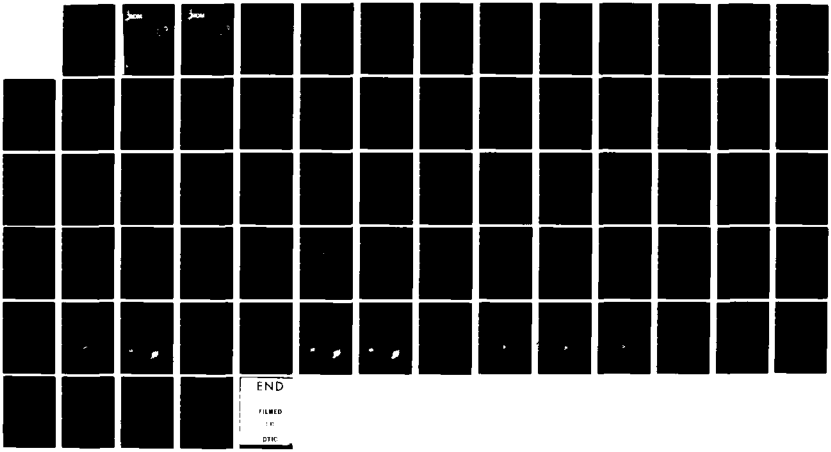
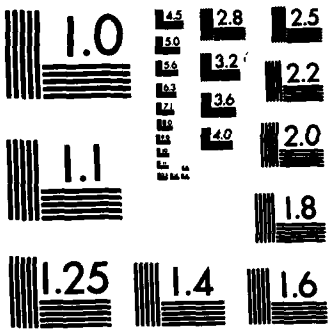
AD-A124 955 AN OPERATION REACTION SYSTEM FOR PLAYER CENTERED MODELS 1/1
REVISION(U) BDM CORP MCLEAN VA J B GILMER 01 DEC 78
BDM/W-78-719-TR-REV DAAG39-77-C-0174

UNCLASSIFIED F/G 12/1 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

THE BDM CORPORATION

AD A124955

83 02 010 015

(1)

THE BDM CORPORATION

7915 Jones Branch Drive
McLean, Virginia 22102
Phone (703) 821-5000

DTIC
ELECTE
FEB 28 1983
H

December 1, 1978

BDM/W-78-719-TR Rev.

John B. Gilmer, Jr.

(1)

Contract DAAG-39-77-C-0174

AN OPERATION REACTION SYSTEM FOR PLAYER CENTERED MODELS

5613A/78W

FOREWORD

The Operation Reaction System (ORS) described here is being used in the Integrated Nuclear and Conventional Theater Warfare Simulation (INWARS), which is under development for the U.S. Army by The BDM Corporation, under Contract DAAG39-77-C-0174. This document will supplement previous descriptions of the ORS in the INWARS Level III Specifications, where it was referred to as The Behavior Generation System (BGS).

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

LIST OF FIGURES

CHAPTER I
THE OPERATION REACTION SYSTEM CONCEPT

The emphasis in a player centered model is on the interactions of the various elements, the total effect of which determine the outcome. Player centered models reduce many modeling problems, since only component processes must be modeled, although this is achieved at the cost of increased software complexity. One of the most difficult component processes to model is the decision making of maneuver entities and command control elements. The Operation Reaction System (ORS) described here is one way of representing these processes which is fast running, relatively simple, and very flexible. It has been used in conjunction with the BDM Corporation's METRIC family of models, which use a hexagonal grid to describe position and movement.

A. BACKGROUND

The heirarchical nature of command relationships results from the practical command limitations of any particular element in the command structure. If large numbers of units are to be controlled, and the C^2 process is to be manageable, a model must similarly limit the number of subordinates and the amount of direction required by the subordinate units. To this end, a design goal has been to incorporate as much at the reaction and operations processes as possible into the lower level units as a means of simplifying the higher level C^2 processing. This also allows a more accurate representation of unit actions in the absence of command control due to enemy action. The Operation Reaction System representation has been designed to meet these goals.

The way in which a unit acts and reacts is dependent on its operation orders. Each unit has an operation order or series of orders which each describe an objective, a mission code for the unit, and an axis which serves to orient the unit with respect ot the enemy. The orders may be issued by superior units, or may be self generated in reponse to a unit's situation.

B. DERIVATION FROM AUTOMATA THEORY

The Operation Reaction System is derived from the theoretical Push Down Transducer, which is itself an elaboration of the Finite State Machine.

1. The Finite State Machine

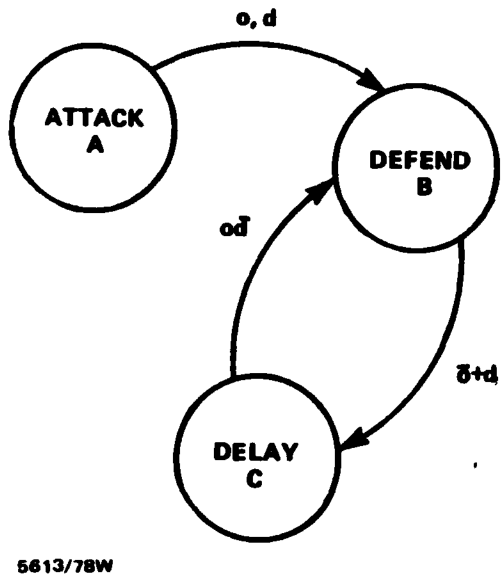
The finite state machine consists of a set of states, rules for transitions between the states for various input conditions, and output definition for each state or transition. When an input is made to the machine, it changes state in accordance with a table which describes the machine. An example of a simple Operation Reaction System mechanism implemented as a finite state machine, is shown in Figure I-1.

In this example, a unit given an initial order to defend with an objective of its own hex would stay in a state, B, representing that mission unless its input indicated a "danger" condition, (d). At that point it would transition to a delay state, C, and would generate a new objective to its rear. The unit would delay until its input condition was "o$\bar{d}$", indicating not in danger, and at its objective, at which time it would return to a "defend" state. A unit ordered to attack would do so until it either was in danger or reached its objective.

The actual implementation would be as a state transition table as shown, a method for generating the input components o and d, and an output mechanism which, in this simple case, is merely the state and whether a new objective is to be generated. The state is thus not only a "mission code," but also governs the parameters used in determining combat effects, situation evaluation, and movement effects. This type of simple implementation is, in effect, a decision table.

2. Push Down Transducer

Note that in the simple Operation Reaction System a unit which is attacking, then perceives danger, looses the initial attack mission when it responds. It 'forgets' what mission it was actually ordered to perform, even after the source of the danger is removed. This problem can be dealt



INPUTS:

0 = AT OBJECTIVE $\quad$ $\bar{0}$ = NOT AT OBJECTIVE
d = IN DANGER $\quad$ $\bar{d}$ = NOT IN DANGER

CURRENT STATE	INPUT			
	$\bar{o}\bar{d}$	$o\bar{d}$	$\bar{o}d$	od
A	A	B	Bg	B
B	C	B	Cg	Cg
C	Cg	B	C	C

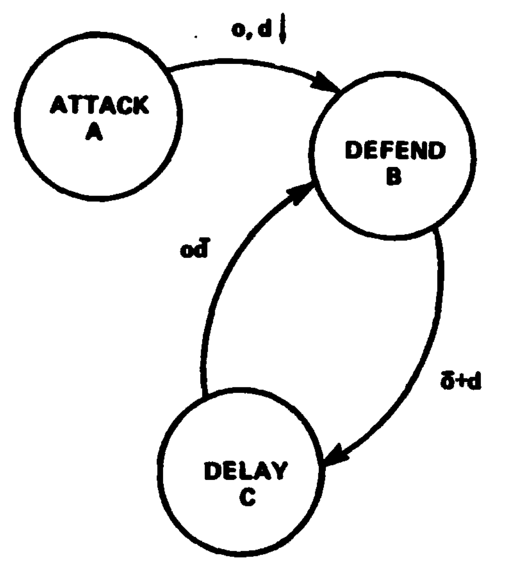
OUTPUT: STATES A, B, C EACH ASSOCIATED WITH PARAMETERS AFFECTING COMBAT, MOVEMENT
g INDICATES NEW OBJECTIVE GENERATED

Figure I-1. Finite State Machine As An Operation Reaction System

with by using a "stack" to save the initial order while dealing with contingencies. Afterwards, the initial order can be restored, and the unit can continue its mission without the intervention of its superior. The addition of a stack to a finite state machine creates a Push Down transducer. A stack can be "pushed" any number of times, with the last order pushed the first to be returned when the stack is "popped". Figure I-2 illustrates a simple Operation Reaction System using this method.

3. Operation Reaction System Output

In both of the examples given, the behavior output is given by the state. This output can be decoupled and made a separate table. This allows the representation of temporary or transition behavior. Figure I-3 shows an example where this is useful, allowing units with delay and attack mission codes to use a "March" operation, or behavior, output while not in contact if they are not at their objective. The full Operation Reaction System is similar to that given in the example, but with modifications made to improve the utilization of computer space by combining inputs in a preliminary step to give a situation code, thus eliminating "don't care" entries from the table.



↓ INDICATES PUSH CURRENT STATE TO STACK.

↑ INDICATES POP STATE FROM STACK IF NOT EMPTY INSTEAD OF TRANSITION INDICATED IN TABLE.

5613/78W

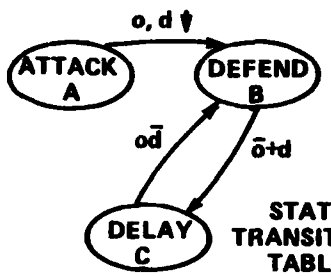
INPUTS:

o = AT OBJECTIVE — $\bar{o}$ = NOT AT OBJECTIVE
d = IN DANGER — $\bar{d}$ = NOT IN DANGER
$o\bar{d}$ = AT OBJECTIVE AND NOT IN DANGER
$\bar{o}$+d = NOT AT OBJECTIVE OR IN DANGER

CURRENT STATE	INPUT $\bar{o}\bar{d}$	$o\bar{d}$	$\bar{o}d$	od
A	A	B	Bg↓	B
B	C↑	B↑	Cg	Cg
C	Cg	B	C	C

OUTPUT: STATES A, B, C INDICATE OPERATION CODE FOR ATTACK, DEFEND, AND DELAY RESPECTIVELY, g INDICATES NEW OBJECTIVE GENERATED

Figure I-2. Push Down Transducer As An Operation Reaction System



INPUTS

o = AT OBJECTIVE
d = IN DANGER
c = IN CONTACT

INPUT

	STATE	$\bar{o}\bar{d}\bar{c}$	$o\bar{d}\bar{c}$	$\bar{o}d\bar{c}$	$od\bar{c}$	$\bar{o}\bar{d}c$	$o\bar{d}c$	$\bar{o}dc$	odc
STATE TRANSITION TABLE	A	A	B	–	–	A	B	Bg	B
	B	C	B	–	–	C	B	Cg	Cg
	C	C	B	–	–	C	B	C	C
OUTPUT TABLE	A	D	B	–	–	A↓	B	Bg↓	B
	B	C↑	B↑	–	–	C↑	B↑	Cg	Cg
	C	Dg	B	–	–	Cg	B	C	C

– INDICATES 'DON'T CARE' INPUT COMBINATIONS;
A UNIT IN DANGER IS ALSO IN CONTACT SO THIS
COMBINATION NEVER OCCURS

OUTPUTS: A = ATTACK
B = DEFEND
C = DELAY
D = MARCH

↑ POP STACK IF NOT EMPTY
↓ PUSH CURRENT STATE ON STACK
g GENERATE OBJECTIVE TO REAR

5613/78W

Figure I-3. Operation Reaction System With Output Tables

CHAPTER II
USE FOR GROUND COMBAT MODELING

The initial application of the Operation Reaction System was to the lowest level ground maneuver entities in the Corps Level Electronics Warfare (CLEW) simulation, in which battalions were the fundamental units, with a 3.5 Km resolution in the hexagonal grid. The Integrated Nuclear and Conventional Theater Warfare Simulation (INWARS) also is using this mechanism, with brigade sized units and 9.5 Km hexes.

In a model, each entity is represented as an Operation Reaction System (ORS), with its state representing an assigned mission, the input being its situation, and the output its behavior.

This representation is advantageous since it allows all of the peculiarities of various missions and modes of operation to be represented by data, the various state and output tables, rather then by complex decision code. This allows the software to be much simpler, faster, and more flexible. Changes in the way missions are executed, situations are evaluated, and actions are taken can be made without modifying the program in most cases; only the data in the tables need be changed.

A. OPERATION REACTION SYSTEM OPERATION

Figure II-1 illustrates the operation of the Operation Reaction System. The cycle of computations shown is repeated separately for all entities. In the two applications used so far, this has been at fixed intervals, with additional cycles upon occurrence of certain events such as movement to a different location.

As a preliminary step, the unit using the ORS must evaluate its situation based on the effects of combat and movement up to that time. This results in a set of situation components, including separate indications of contact with enemy units, danger of being flanked, own casualty or supply status, meeting engagement conditions, nuclear or chemical status, etc. Direct application of so many input variables would require unreasonably

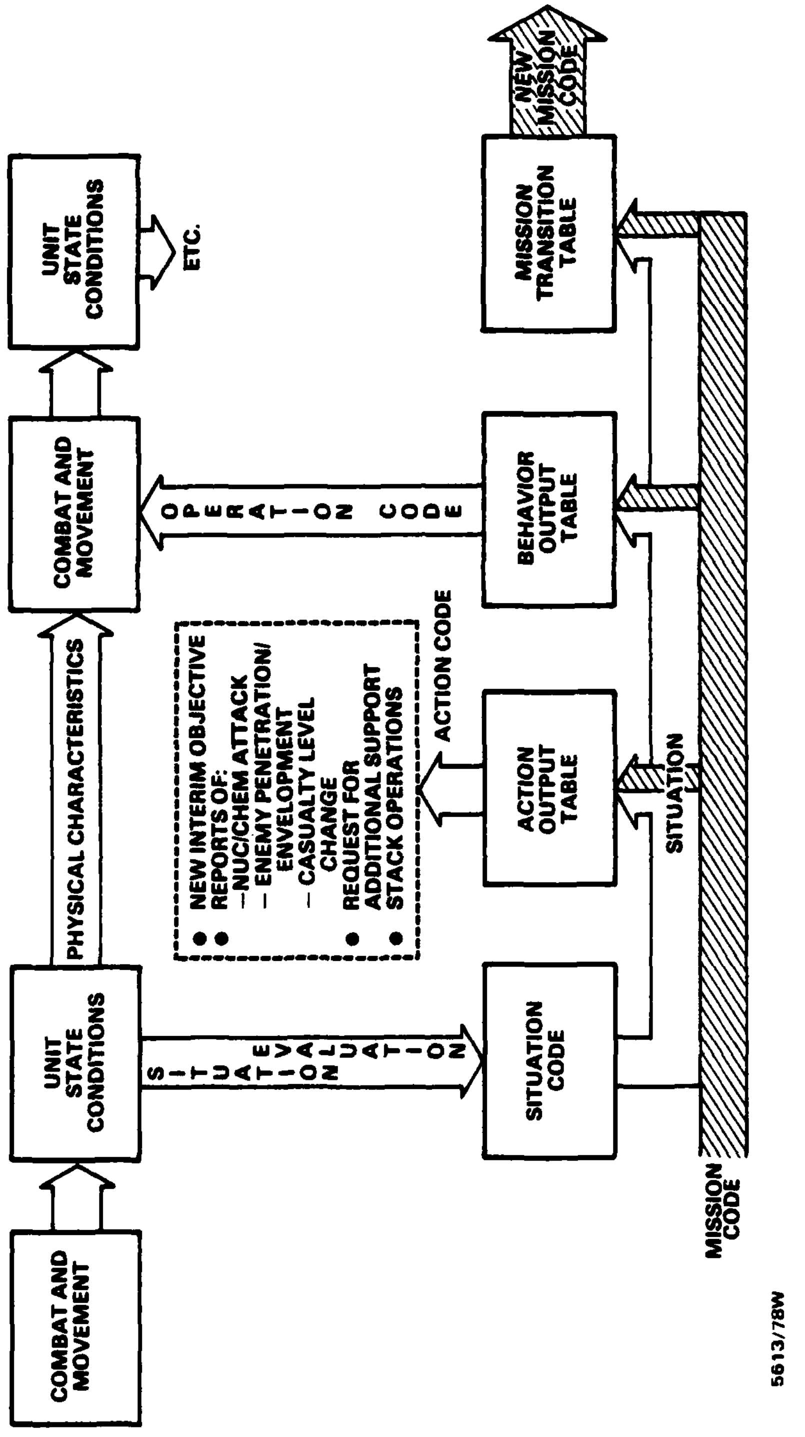


Figure II-1. Operation Reaction System Overview

large tables, so a new table and/or functions are used to reduce the number of possible input combinations. All input component combinations having similar operational effects are given a single situation code, which then becomes an input to the Operation Reaction System proper.

The Situation Code and Mission Code of the unit's current operation order are used as inputs to three tables. The first is used to look up an action code, which in turn indexes into a table giving the actions which are to be initiated by the unit, including stack push operations, requests for support, generation of new objectives, etc. The second table gives an operation code which determines parameters affecting combat, movement, and situation evaluation for the next cycle of the physical processes. A third table yields a new mission code which will replace the previous code, although in many cases it will be the same. If a stack push operation occurs, the old mission code is saved, so that the new value does not destroy the previous order.

When a stack push occurs, a new operation order is created for the unit. The old order is linked to the new one, with a contingency code indicating the conditions under which the stack is to be popped and the old order restored. The "pop" operation occurs after the situation code is computed, but before any table lookups. Thus if the contingency code causes the stack to be popped, the new order's mission code is used in the ORS cycle rather than that of the discarded order. The order to 'push' the stack must also indicate whether the old objective is to be saved in absolute or in relative form. In the latter case, an absolute objective is computed for the same position relative to the unit as before the push was executed. The Unit then continues to undergo combat and movement processes until it again repeats the Operation Reaction System cycle.

B. DEFINITIONS

The Operation Reaction System state, input, and output are defined as follows:

(1) State = Mission - The states in the ORS correspond to missions that are to be performed by the unit, such as hasty attack, delay, march etc.

(2) Input = Situation - The input to the ORS will be a code corresponding to the unit's situation. The encoding will include such factors as a unit's effectiveness status, combat status, the enemy threat, and nuclear/chemical environment. This is most easily represented by a table which gives the input code as a function of all the various aspects of the situation.

(3) Output = Behavior and Actions - The outputs of the ORS will be in two categories:

1. Behavior parameters - This will be an operation code which is used to look up a set of parameters governing combat and movement. These will include the operational inputs to the attrition equation, criteria used in evaluating movement options, effectiveness breakpoints, delay times, internal disposition, and others.
2. Action Code - This code indexes into a table containing flags which initiate actions such as pushing of the stack, calls for artillery or close air support, initiation of messages, generation of new objectives, or any other single actions appropriate for the unit.

A detailed example of an Operation Reaction System is given in Appendix A. Examples of operation from CLEW are given in Appendix B.

C. APPLICATION OF THE OPERATION REACTION SYSTEM TO COMMAND/CONTROL PROCESSES

A hierarchical command structure can be implemented using the Operation Reaction System at each level, applying it to the players as well. It

is also possible to use such a system for lower level players, while using other simulation methods or man in the loop to represent the higher command levels. The Operation Reaction System can be applied to the operation of the players, or C^2 elements, as well as the maneuver entities by making the following adaptations:

(1) Inputs - The players must consider more factors than the smaller entities. In addition, a composite measure of effectiveness is needed. The size of the situation table can be reduced by combining some elements before the table look up step.

(2) Actions - in addition to the simple actions performed by the basic entities, C^2 elements must be capable of allocating artillery support, initiating reinforcement from reserves, and reallocating forces. Most important, they must be capable of planning a position or objectives for their subordinate units, and issuing the necessary orders.

(3) Input of Plans from Higher Echelons - when the higher echelon command develops a plan it will pass a stack of operation orders to its subordinates, which will attempt the execution of various types of attacks, defense, etc. The actual planning at each command level is accomplished as each operation requiring planning is popped from the stack as the sequence of objectives or phases are reached.

1. <u>C^2 Player Operations</u>

The following is a possible list of operation codes:

(1) Defense: This is the basic operation type for a defensive posture, it assumes the aggregate unit will maintain a cohesive position within a given region.

(2) Delay: This is like the defense except that individual subordinate entities will make tactical withdrawals if faced with large force ratios, and will attempt to trade time for space.

(3) Main Attack: This is the basic attacking posture; usually will employ available forces in two echelons in a fairly narrow sector.

(4) Secondary Attack: This form of attack will employ a smaller proportion of the aggregated unit in reserve, and will normally use a larger sector width. Subordinate entities use a corresponding operation code to minimize losses

(5) Breakthrough: This attack operation is used to achieve maximum concentration.

(6) Reserve: This is used for units which are uncommitted.

2. <u>Actions</u>

These actions would be initiated by the players:

(1) Generation of a New Objective: This generally corresponds to the same action of the basic entities, except that objectives are described in terms of regions rather than hexes. An objective region could be a center hex, an axis of orientation, and a sector width. Generated objectives would usually adjoin or overlap previous ones.

(2) Call for artillery support or attack helos: The unit could request additional allocation of artillery or attack helicopters from its superiors.

(3) Call for close air support: In addition to the requests from individual entities, the C^2 player could add weight to the priority requested.

(4) Call for reinforcement. This initiates a message to the player's superior requesting additional ground combat assets.

(5) Report initiation: This generates a situation report.

(6) Push operation: This directly corresponds to similar actions of the basic entity ORS.

(7) Reserve Commitment: This action specifies the commitment of a specified force from the player's reserves or second echelon.

(8) Force shift: This is the shifting of assets from a less threatened area to a more threatened area.

(9) Change Axis: The player's axis of operations may be changed, particularly in those cases where strong flanking forces exist.

(10) Plan: This flag initiates the planning process.

3. Planning

At some stage in the planning process generally defined regions must be translated into particular hex objectives for particular units. This is done in the planning process. The planning process would take place in two steps:

a. Position Definition

In this step the C^2 player chooses a series of hexes of objectives within the region. For defensive operations they are chosen to span the sector width with no gaps. For offensive operations they would be chosen around the center of the region with an emphasis on key terrain. A template type assignment for certain cases could also be used, such as for assembly areas in preparation for an attack.

b. Unit Assignment

The units are assigned to particular objectives. Combinations which are feasable can be scored, and the units assigned to hex objectives and given appropriate sector widths in accordance with the best combination.

CHAPTER III
IMPLEMENTATION

A. STEPS IN IMPLEMENTATION

The following steps are used to develop the data and code for implementation of this method:

(1) Coding of the ORS mechanism - since this code is independent of the operation implementation, structures, and parameters, it can either precede or parallel the following steps.

(2) Description of Actions - those actions which must be represented must be described so that code to implement them can be written. They need not be associated yet with the conditions that require them.

(3) Description of Situation - the inputs to the ORS must be described so that code to recognize these conditions can be written.

(4) State Assignment - a set of mission codes which will describe the various modes of operation of the entities must be described. These will correspond to the states. The rules for state transitions then must be described in terms of the situation description of 3 above. These rules must also include the pushing and popping of the stack.

(5) Output Definition - the outputs associated with the various states or tansitions must be described in terms of the output parameters and actions, and the required inputs to drive them.

(6) Input Table Construction - a table can then be constructed to translate the situation aspects into the codes necessary to define all of the state transitions and outputs.

(7) State and Output Table Construction - these tables can be defined given the inputs and the results of steps 4 and 5 above. This completes the ORS design.

B. AUTOMATED AND LANGUAGE PROCESSING AIDS

The task of filling out the tables for an Operation Reaction System implementation can be difficult, particularly for applications requiring many states. It is possible to use automated methods for doing this which will both simplify the design of the tables and provide meaningful documentation. A body of software to accomplish this would essentially be a simple compiler. Appendix C gives an example of an input language to Operation Reaction System specification.

C. HAZARDS AND CYCLES

It is possible for the ORS implementation to include hazzards and cycles which will result in inappropriate behavior. In the former case, one input occurs before another, causing a different reaction than if both occurred at the same time. While this is often desired, a hazzard can occur if this is inadvertent. A cycle occurs if, for a particular input condition, the state cycles among a number of states. Since the process of designing the ORS tables is analogous to synchronous digital design techniques, methods developed for ensuring hazzard and cycle free design for the latter can be applied.

D. ORDER IMPLEMENTATION

The orders appropriate for the units are attached to them. During the scoring of unit assignments, estimates for time to perform the mission or capability may be developed which could be passed to higher C^2 elements.

E. CONCLUSION

The ORS can be applied to the modeling of the conduct of operations, allowing significant savings in software development effort and enhancing realistic treatment of a unit's actions. In addition, changes or additions

to the set of operations or their transitions can be accomplished by changing the data in tables rather than by modification of the code. This makes the combat process more accessible to the user and reduces the cost of implementing changes to represent new doctrine or tactics. Another aspect of this structure is that the implementation of complex plans generated by higher echelons can be implemented by passing a unit a new stack of operations to be executed. This allows significant simplification of the software.

CHAPTER IV
EXPANSION AND ENHANCEMENT POTENTIAL

The basic Operation Reaction System has potential for further development which will enhance its capability to represent decisionmaking. The features discussed here are particularly useful in the representation of the player elements.

A. MULTIPLE DOCTRINES

Different doctrines can be used without modifying the basic Operation Reaction System by expanding the number of mission codes and the size of the tables. Different nationalities, sides, or types of units could each have a separate set of mission codes. The actual finite state machine tables of the Operation Reaction System would constitute a set of sub-machines, each consisting of a set of states from which there are no transitions to any state in another sub-machine. Under these conditions, there is no software difference from the basic ORS described earlier, although the tables would be defined differently.

B. CONTINGENCY ORDER STRUCTURE

This feature allows a structured set of orders to be passed which will describe an operation to be executed, with alternative orders which cover specific contingencies. This is an elaboration of the "stack" previously discussed. This allows the ORS system to be overridden or modified to deal with particular or unusual circumstances.

1. Operation Order

The basic operation order consists of a mission code, an objective, and an axis of operation which serves to orient the unit with respect to the enemy. In addition, there are links to contingency orders. Each link contains not only the pointer to the contingency order, but also a

contingency code which indicates the conditions under which the contingency is recognized, and a disposal code which indicates what is to be done with the linked order if some other contingency is recognized instead. The operation order structure thus takes the form of a tree, as illustrated in Figure IV-1.

2. Contingency Recognition

The contingency code identifies the situation in which a particular contingency operation order is to be adopted. It defines a set of situation codes or situation components which constitute that contingency, in the form of a list.

3. Disposal Conditions

When a contingency is recognized, and the contingency operation order is adopted, the old operation order is disposed of. It is also necessary to dispose of unused contingencies. In the simplest case, these, like the old operation order, are thrown away. But it can be useful to define a different type of disposal code which indicates that the contingency is to be passed to the new order. This allows a particular contingency, which is to be considered regardless of the phase or state of the operation execution, to be present in only one copy rather than as a separate contingency attached to every other operation order in the structure. In addition, it is useful to define a disposal code which will keep a contingency current when the stack is pushed. It would be linked to the new order as well as the old one. This, in turn, requires an additional 'no disposal' code, so that when the new operation is returned the contingency order will not be lost. Figure IV-2 illustrates a possible resulting order structure.

4. Template Orders

All of the operation orders discussed to this point have been detailed with an objective and axis which apply to a specific unit. In order to simplify the construction of order structures and save space, a template form of order can be used. This format gives an objective with respect to the unit's current location and axis, and can thus be applied to many different units. A template order could be linked into the operation

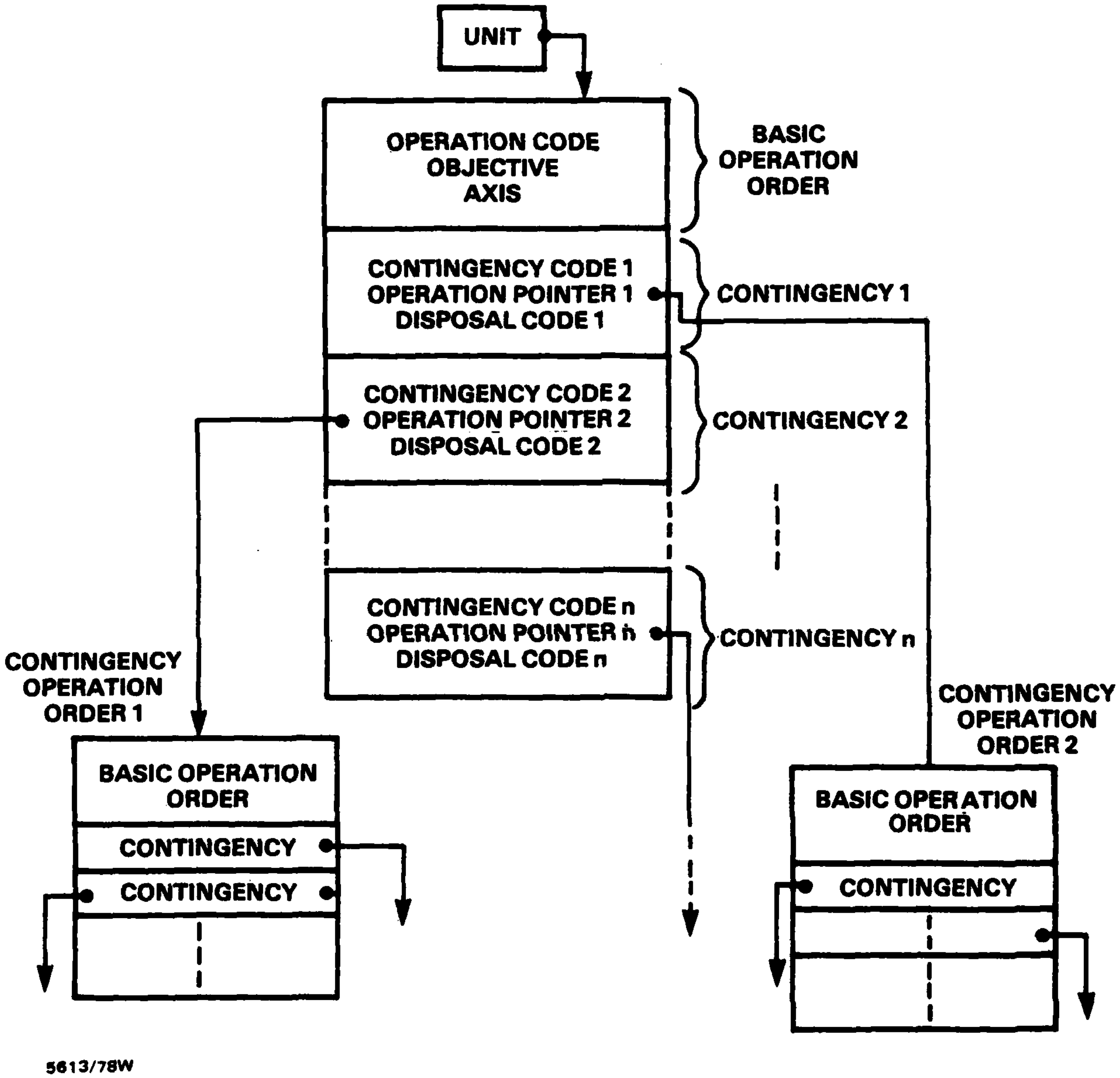
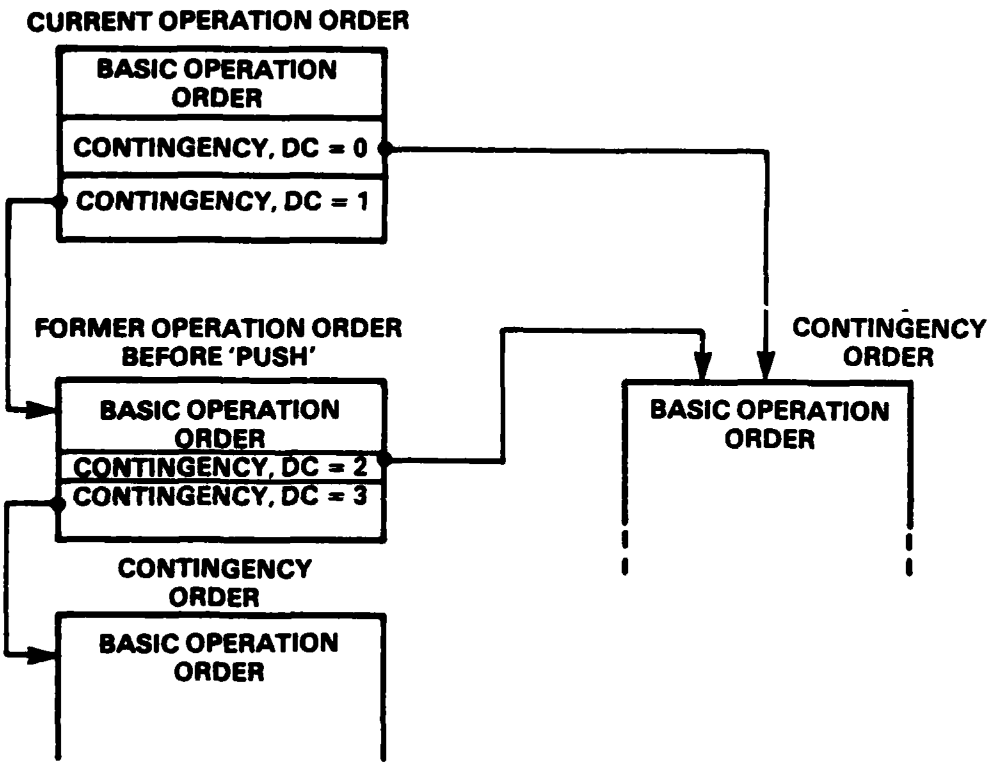


Figure IV-1. Operation Order Structure



KEY: DC = DISPOSAL CODE:
0 = DO NOT DESTROY
1 = RETURN AND DISPOSE NORMALLY
2 = PASS TO CURRENT ORDER ON A 'PUSH'
3 = PASS TO CURRENT ORDER ON A 'POP'

5613/78W

Figure IV-2. Use of Contingency Disposal Codes

order structure. If its contingency arises, the order would then be detailed for the particular unit, thus creating a "normal" operation order in its place. Thus, many different units could have contingency pointers to the same template order structures, saving space since separate orders would be detailed only if needed, making significant computation and space savings. Figure IV-3 illustrates.

C. DISAGGREGATION OF OPERATION PLANNING

If the Operation Reaction System is applied to the operation of the command/control elements, there must be a method for implementing the aggregated unit's order as a set of orders to subordinates. The use of Template orders, described above, allows a simple representation of this.

The operation code of the C^2 element would have associated with it an implementation order structure, as illustrated in Figure IV-4. This consists of a set of roles, each to be associated with a particular subordinate unit. Each role has an operation order chain of template format orders. When the unit implements planning for its operation, as initiated by an ORS action, the operation order structures are passed to the subordinates, with the initial order detailed for the particular location and axis associated with each role. Other orders in the structure remain in template form and are detailed if needed.

D. OTHER APPLICATIONS

The Operation Reaction System can be applied to other types of models also. In the INWARS model, it is also used for air units, with different states used for missions such as close air support, interdiction, air superiority, etc. It is also possible to use it in modeling the performance of tasks, such as the performance of the operation of ESM equipment. Different codes could represent modes of operation, including scanning for signals, listening on a particular frequency, etc.

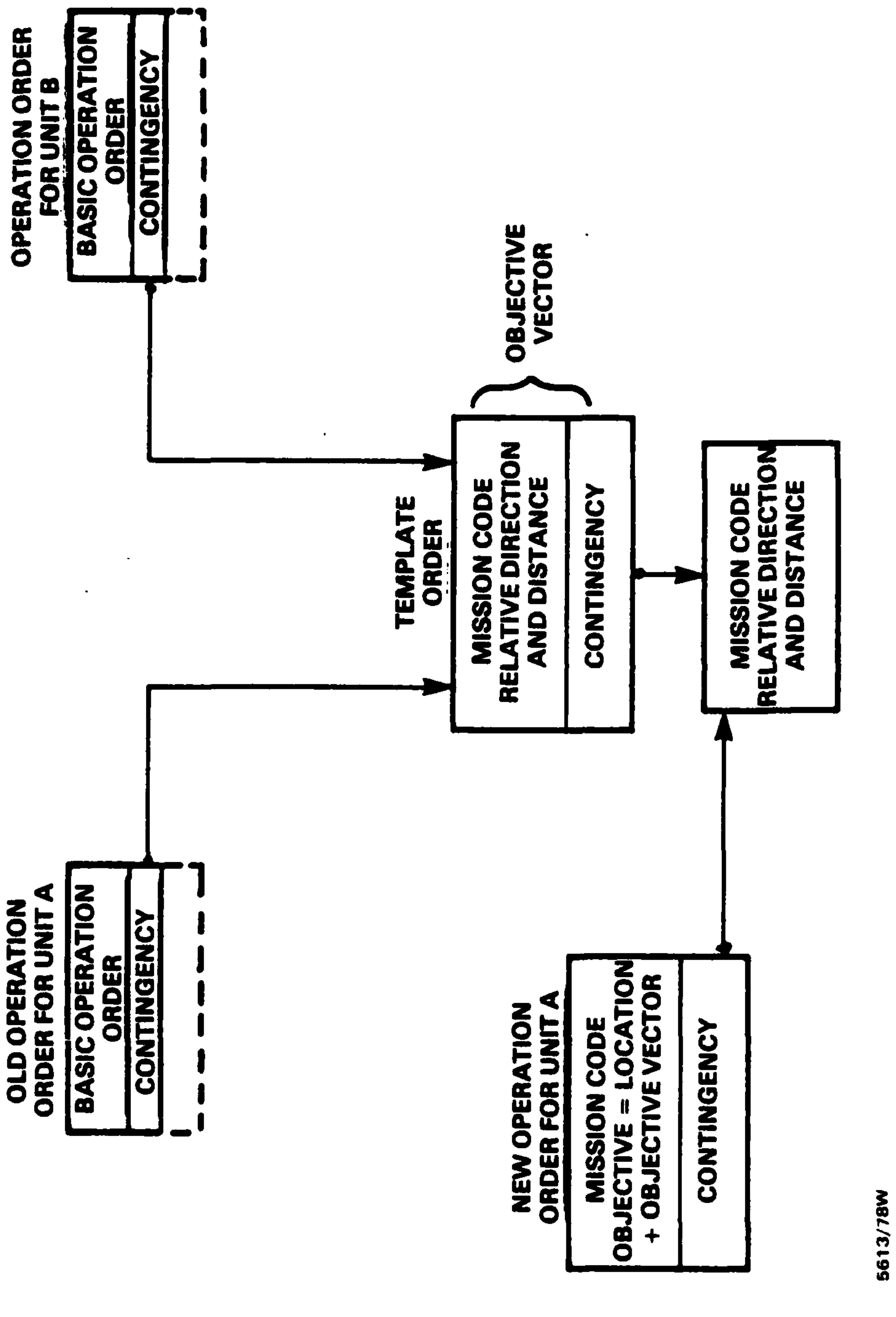


Figure IV-3. Use of Template Operation Orders

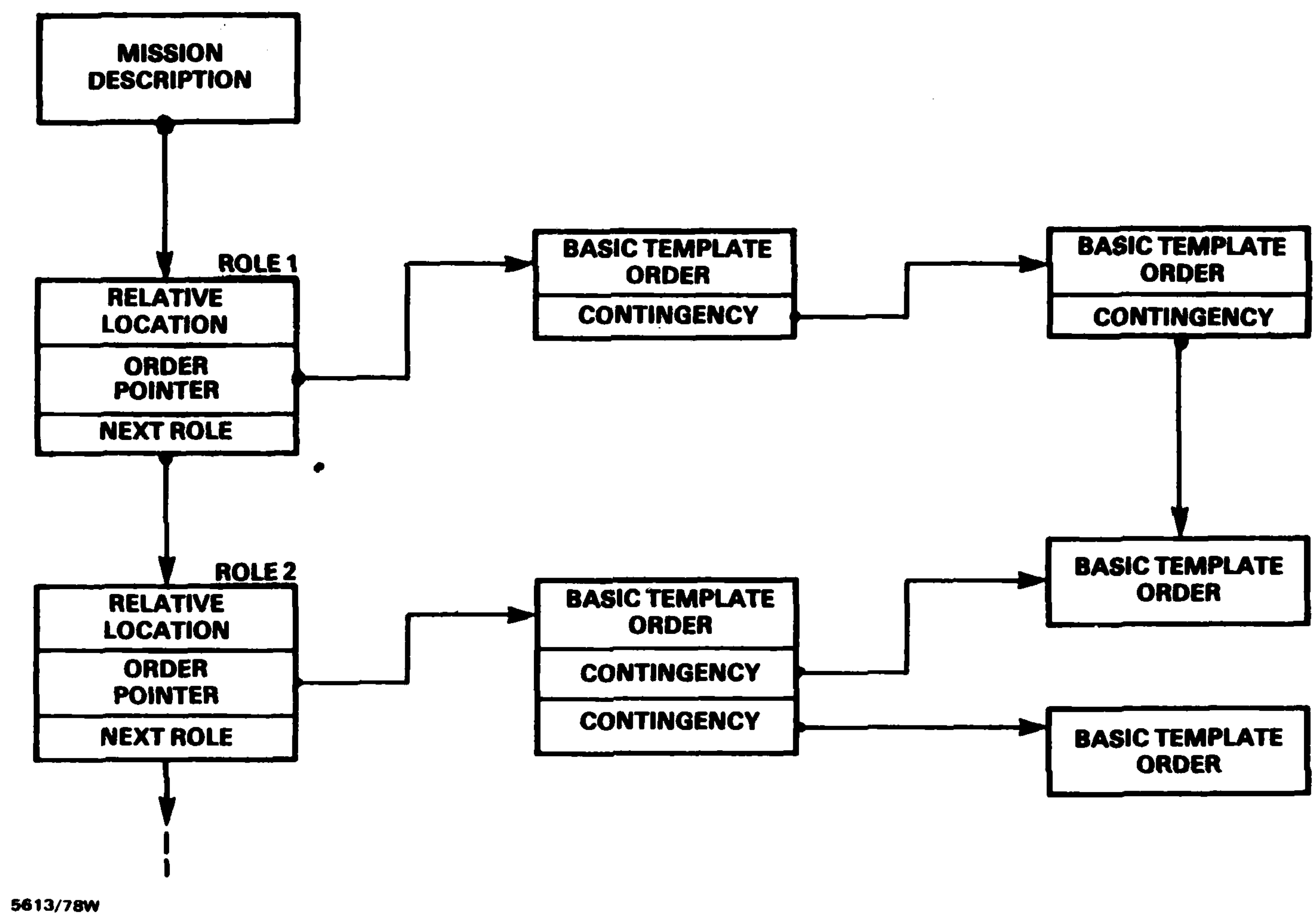


Figure IV-4. Disaggregation of Operation Planning

CHAPTER V
CONCLUSIONS

A. BENEFITS AND LIMITATIONS

The Operation Reaction System as described has the following benefits and limitations:

(1) Allows Significant Savings in Software Development: The code to implement the ORS is much simpler and more general than corresponding if then type decisionmaking implemented in software.

(2) Enhances Realistic Treatment of Unit Actions: The reaction and adjustment processes inherent in the ORS implementation allow realistic representation of the operation of maneuver entities. This is particularly true when it is desired to show effects of loss of command/control, where units must continued to operate reasonably, if in uncoordinated fashion, despite lack of direction from above.

(3). Facilitates Changes to Operations: The representation of operations and doctrine can be changed with data by modifying the ORS tables, rather than by changing compiled code.

(4) Provides a Simple Mechanism to Play Alternative Doctrine: This is implemented with table modification or additions.

(5) Is User Oriented: The manipulation of the decisionmaking logic is within the capability of the user, since no code changes are involved.

(6) Allows for Significant Enhancement Possibilities: These include the application to hierarchical command structures and other types of applications.

(7) Table Construction Difficult Without Implementation Aids: The construction of tables for the ORS is time consuming and difficult, particularly for large numbers of operation codes and situations. This problem could be eased by a processor for designing the tables.

(8) Doctrinal Reaction, not Innovative: The reaction of units to a particular situation is dictated by the tables, and does not allow for innovative or unusual reaction to the situation by units.

(9) Lack of Resolution in Situation Components: The components which make up the situation must be relatively few in number and quantized roughly to reduce the number of table entries. This results in an inability to distinguish fine points or unusual circumstances. Part of this problem may be made up for by the fact that effects should average out over large numbers of units. But in following a particular unit, unusual or unrealistic reactions can sometimes be observed. The practical use of more detailed situation components, which would help solve this problem, depends on the use of implementation aids and computer space.

B. RESULTS FROM THE CORPS LEVEL ELECTRONICS WARFARE SIMULATION

A simpler Operation Reaction System than that detailed here was used in the CLEW simulation. It had all of the features except the 'push' capability and contingency operations. The stack, inserted by the man in the loop as a set of orders, was 'popped' upon arrival at an objective not occupied by enemy units. The ORS was used for all maneuver entities, with higher echelons played by a man-in-the-loop commander. Appendix B gives examples from operation of the CLEW model.

The ORS allowed CLEW to run for longer lengths of time relative to the resolution between man-in-the-loop interventions then had been possible in a previous version of CLEW without the ORS. It was also possible to use fewer orders, since unit behavior was less dependent on direct supervision, of the units. It was found that the combat behavior of units was reasonable, even with large numbers of units interacting. Some problems were encountered in making units follow particular roads, since the entities often exercised independent decisions to move off the desired road in order

to avoid friendly units ahead. This was more of a problem with the march operation parameters and perception process than with the ORS structure itself, however.

C. SUGGESTIONS FOR FURTHER RESEARCH

The Operation Reaction System has shown that it is a considerable aid in realistically representing ground combat operations. The following areas of research could significantly extend or improve its capabilities:

(1) Automated ORS Table Generation: This would make the use of the ORS much more responsive to the users, would allow the use of much more complex tables, and would serve to generate documentation. The use of language processing techniques such as lexical analysis and parsing should be applied to build a ORS compiler' to realize this. The techniques used for automated digital design can also be applied.

(2) Heirarchical Operation Planning: The use of the Operation Reaction System in a hierarchical command structure has not been fully developed, although elements of this application have been discussed. The larger number of situation components and considerations would make the use of automated table generation much more necessary. The use of choice of operation to fit circumstances rather than just reaction is also an area to be explored.

(3) Stochastic Operation: In the Operation Reaction System discussed, there has been a deterministic choice of action. The use of a stochastic mechanism might be useful in more accurately representing the possible effects of command initiative or degradation.

(4) Adaptive Input Processing: The Operation Reaction System currently modifies threshholds used in the situation analysis to account for the current operation code. An extension to an input filtering concept would allow the perception process to be more appropriate for particular operations. This would not only allow

more detailed situation inputs, but could reduce processing time since unnecessary elements of the situation would not be computed.

APPENDIX A
OPERATION REACTION SYSTEM EXAMPLE

The following are the mission/operation codes and actions.

1. <u>Mission and Operation Codes</u>

(1) Prepared Defense: used when a unit has been in place in a hasty defense for a specified time

(2) Hasty Defense: basic defensive posture

(3) Delay: used to trade space for time; decreases attrition rate

(4) Withdraw: used by units to break contact, or initiated by the ORS due to loss of effectiveness

(5) Hasty Attack: Basic attack posture for Red, also used for meeting engagement

(6) Coordinated Attack: Basic attack posture for Blue

(7) Breakthrough: This operation allows considerable massing and high attrition rates at some cost in attacks vulnerability

(8) Reconnaissance: Used by advanced elements by Red.

(9) March: used for non combat movement

2. <u>Action Codes</u>

(1) Generation of a new objective: this causes a unit to consider changing its objective. The action gives a distance, either ahead or behind the unit, for the new objective. The hex at that distance along the unit's axis of operation then may become the new objective. If the distance is only one hex, however, the actual objective is chosen using the movement scoring mechanism. The objective of a unit is changed only if the unit is not already moving on a path within 60° of the direction to the desired direction.

(2) Call for artillery support: This action specifies a level of GS artillery support desired from higher echelon assets.

(3) Call for close air support or attack helos: similar to artillery, but these requests are simply relayed to higher echelon elements by the unit.

(4) Call for reinforcement: This action influences the assessment of the situation, and may cause planning, the release of resources, or the sifting of assets from one entity to another.

(5) Report of critical situaton: This action will initiate a message to the unit's superior.

(6) Push stack: This flag will cause the previous situation to be saved on the stack, along with information on the conditions under which it should be restored.

(7) Pop Stack: If there are states saved on the stack, the top one will be restored as the state, or operation, of the ORS.

3. Example:

Figures A-1 through A-5 illustrate a preliminary example of how the Operation Reaction System might be implemented in INWARS for Brigades and Regiments. This example is for illustration only. The data and format of the tables will be revised and developed in accordance with the process described in the text. Figure A-6 illustrates this example.

A unit initially is at its assigned objective, and is in a hasty defense mission. It has suffered no significant casualties, and has an effectiveness degradation state of 0, indicating no impairment. No further operations are in the unit's operation stack awaiting execution.

(1) Time step 1: During the unit's perception process, enemy units are found in adjacent hexes, but do not present a flanking threat. The situation code tables (Figure A-1) is used to find the appropriate situation code, 16, for the situation of not being danger of being flanked, adjacent to enemy units, not being an immediate chemical or nuclear victim, at the unit's objective, and having no degradation of effectiveness. The action table (Figure A-2) is then entered for the situation code 16 and the mission code of 2 for Hasty Defense; an action code of 12 is found. The action code table (Figure A-3) indicates a low priority request is made for air and artillery support to take advantage of the targets presented. The pop operation stack flag

is ignored since no further orders are in the stack. The Operation Behavior Table (Figure A-4) is also entered with the mission code of 2 for Hasty Defense and the situation code of 16. A blank entry (zero in the software implementation) indicates the unit will behave normally in a hasty defense. The Mission Transition Table (Figure A-5) entered for Hasty Defense and situation code 16 indicates no change.

(2) Time step 2: Since the previous interval, an enemy unit has moved into a position to threaten the unit's flank. The situation code of 17 results. The action table yields a code of 1, which indicates the operation stack is to be pushed, saving the hasty defense mission. The unit will also generate an objective one hex back, so that movement will not be toward that objective. The hex will be chosen on a basis of various threat, cover, cohesiveness, etc. considerations. Requests are made for air and artillery support. The operation behavior and mission transition tables indicate a mission code of 3, indicating the unit is now delaying back the one hex to the new objective.

(3) Time step 3: At this time one enemy unit has attacked into the same hex. The unit has not yet withdrawn out of the hex back to its new objective, and remains in danger, resulting in a situation code of 6. For the Delay mission, action code 10 is found indicating artillery and air requests and the generation of a new objective. The latter action is ignored since the unit is already moving in an appropriate direction. Blanks in the Operation Operation and Mission Transition tables indicate continuation of the Delay operation.

(4) Time step 4: The unit has now arrived at its objective hex, but the flanking enemy unit has also moved into that same hex resulting in a meeting engagement. The resulting attrition has increased the unit's effectiveness degradation level to 1, meaning it is now marginally effective. The situation code of 10 and mission code of 3 yield an action code of 3, which indicates

high priority air support and artillery requests, and calls for reinforcement. A special situation message to Corps is also called for. A new objective one hex further to the rear is generated. The operation behavior table gives a code of 5, indicating a hasty attack posture for the meeting engagement condition, although the mission transition table indicates a continuation of the delay mission.

(5) Time step 5: The unit has withdrawn from the hex in which the meeting engagement took place, and has now reached its objective. No enemy units are in the hex or present a flanking threat. Effectiveness degradation is still at level 1. Situation code 18 is found; action code 12 results from that and the delay operation status. Artillery and air support are requested, and a stack pop is made, restoring the unit's original mission code of hasty defense. The operation output code is 2, or hasty defense, and the operation transition result is ignored since the stack supplied the next mission code.

NUCLEAR VICTIM	AT OBJECTIVE	EFFECTIVENESS DEGRADATION	NOT IN DANGER				IN DANGER		
			nc	adj	hex	mtg	adj	hex	mtg
no	no	0	1	2	3	4	5	6	7
no	no	1	1	8	9	10	11	9	10
-	no	2	12	13	14	14	14	14	14
no	yes	0	15	16	3	4	17	6	7
no	yes	1	15	18	9	10	19	9	10
-	yes	2	20	13	14	14	14	14	14
yes	no	0	21	22	9	10	23	9	10
yes	no	1	21	24	14	14	25	14	14
yes	yes	0	26	27	9	10	28	9	10
yes	yes	1	26	29	14	14	20	14	14

Figure A-1. Situation Code Table

ACTIONS

MISSION	SITUATION (3 rows for 0-9, 10-19, 20-29 respectively) 0	1	2	3	4	5	6	7	8	9
1. Prepared Defense	0	17	5	8	8	9	9	8	9	2
	2	0	17	2	3	4	12	9	12	9
	4	14	15	16	16	16	14	16	16	16
2. Hasty Defense	0	17	5	8	2	1	2	2	9	2
	2	1	17	2	3	4	12	1	12	2
	4	14	15	16	16	2	14	16	2	2
3. Delay	0	17	5	10	2	10	10	2	9	2
	3	9	17	2	3	4	12	1	12	2
	4	14	15	16	2	2	14	16	2	2
4. Withdraw	0	17	5	9	11	10	10	11	9	3
	3	9	17	11	3	4	1	1	11	11
	4	13	3	3	3	3	13	11	11	11
5. Hasty Attack	0	17	5	8	8	5	8	8	11	11
	11	11	17	5	3	4	12	12	12	12
	4	13	15	15	16	16	13	15	15	16
6. Coordinated Attack	0	17	5	8	8	5	8	8	11	11
	11	11	17	5	3	4	12	12	12	12
	4	13	15	15	16	16	13	15	15	16
7. Break-through	0	17	8	8	8	8	8	8	11	11
	11	11	17	5	3	4	12	12	12	12
	4	13	15	15	16	16	13	15	15	16
8. Recon	0	17	8	8	8	8	8	8	11	11
	11	11	17	5	3	4	12	12	12	12
	4	13	15	15	16	16	13	15	15	16
9. March	0	0	4	18	18	12	18	18	12	18
	18	12	0	18	18	4	12	12	12	12
	4	13	2	2	2	2	13	2	2	2

Figure A-2. Action Table

ion de	push	pop	gen obj fwd	gen obj back	consider move	arty req	air req	relnf req	send msg
	X			1		1	1		
	X			1		2	2	1	1
				1		3	3	3	1
		X							
							1		
					X	1	1		
			1						
						1	1		
						2	2	1	
0				1		1	1		
1				1		2	2	1	1
2		X				1	1		
3	X								1
4									1
5						1	1		1
6						2	2	1	1
7	X								
8		X		1		2	2		1

Figure A-3. Action Code Table

MISSION	SITUATION CODE (3 rows for 0-9, 10-19, 20-29) 0	1	2	3	4	5	6	7	8	9
1. Prepared Defense		9			5			5		3
	5		9	3	4					
2. Hasty Defense		9			5		3	5		3
	5		9	3	4				3	3
					3	3		3	3	3
3. Delay		9			5			5		
	5		9		4	2	2		2	
	2									
4. Withdraw		9			5	2	2	5		
	5		9			2			2	
	2									
5. Hasty Attack										
					4					
		2						2		
6. Coordinated Attack										
					4					
		2						2		
7. Breakthrough										
					4					
		2						2		
8. Recon										
					4					
		2						2		
9. March			2	5	5	2	5	5	2	5
	5	2		2	4	2	2	3	2	1
	2	2	2	3	2	3	2	2	3	3

Figure A-4. Operation Behavior Table

MISSION	SITUATION CODE (3 rows for 0-9, 10-19, 20-29)									
	0	1	2	3	4	5	6	7	8	9
1. Prepared Defense		9								3
	3		9	3	4					
2. Hasty Defense		9			3		3	3		3
	3		9	3	4			3		3
					3	3			3	3
3. Delay		9								
			9		4	2	2		2	
	2									
4. Withdraw		9				2	2	3		
			9			2			2	
	2									
5. Hasty Attack										
					4					
		2						2		
6. Coordinated Attack										
					4					
		2						2		
7. Breakthrough										
					4					
		2						2		
8. Recon										
					4					
		2						2		
9. March			2	2	2	3	3	3	2	4
	4	3		4	4	2	2	3	2	3
	2	2	2	3	2	3	2	2	3	2

Figure A-5. Mission Transition Table

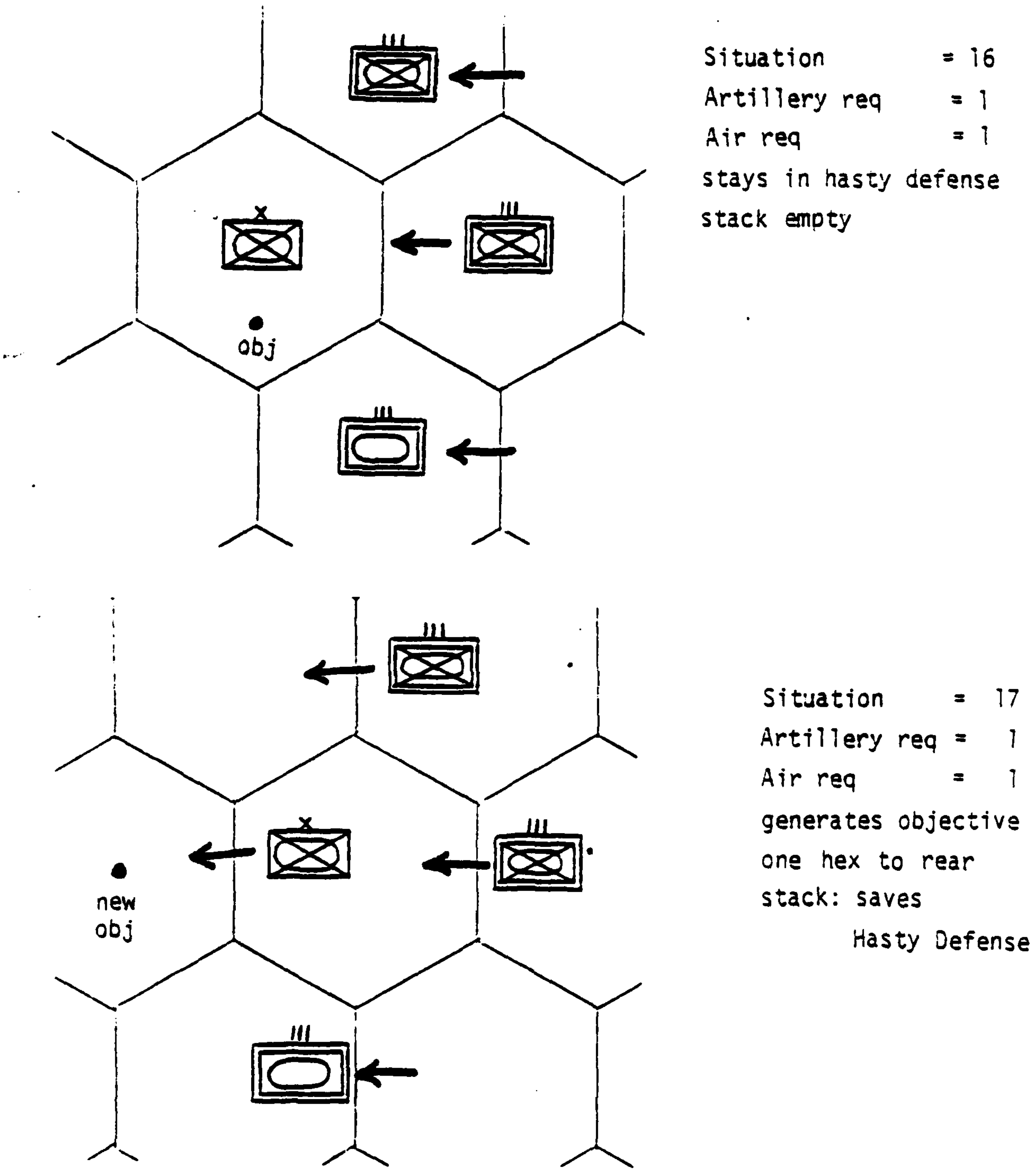
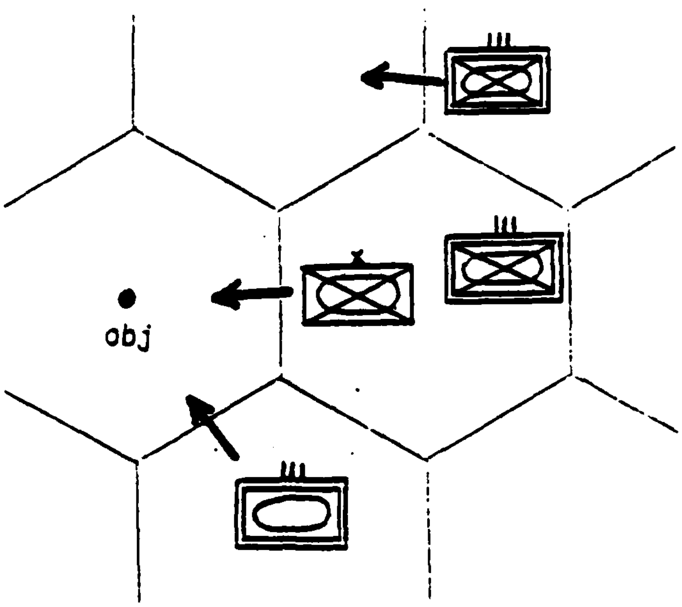
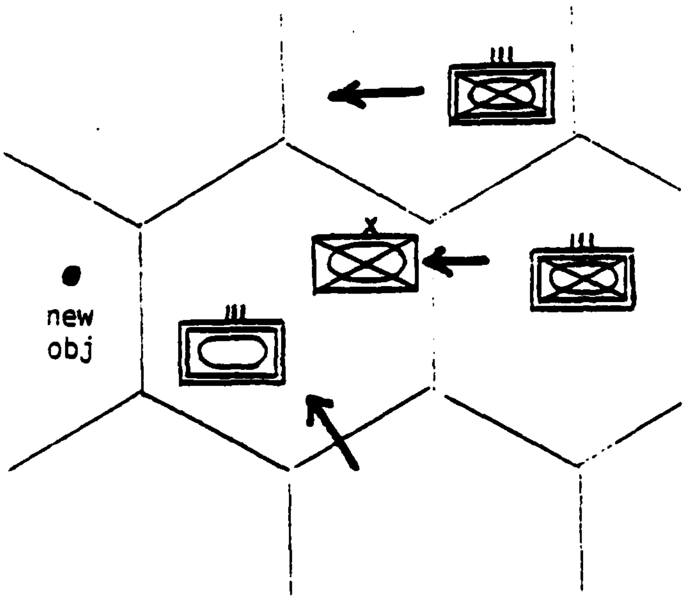


Figure A-6a. Example of Operation Reaction System Operation, Time Steps 1 and 2

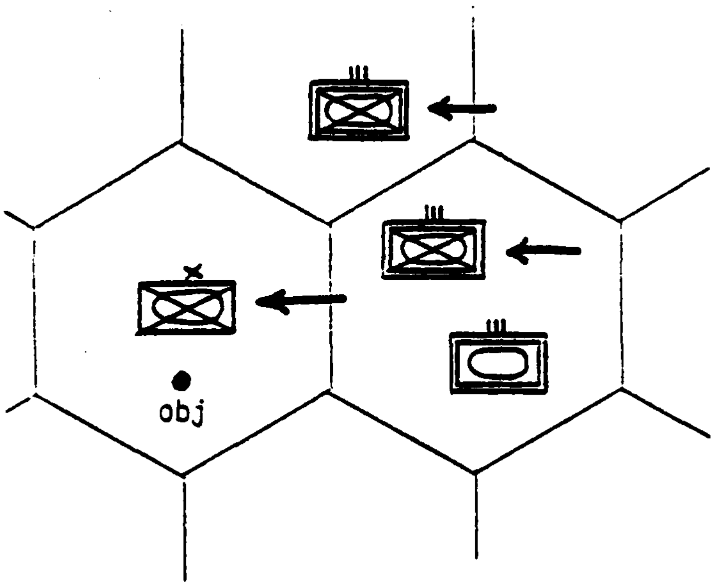


Situation = 6
Artillery req = 1
Air req = 1
moving to new objective
stack: Hasty Defense



Situation = 10
(meeting
engagement)
Artillery req = 3
Air req = 3
Reinforcement requested
generates objective
one hex to rear
stack: Hasty Defense

Figure A-6b. Example of Operation Reaction System Operation, Time Steps 3 and 4.



Situation = 18
Artillery req = 1
Air req = 1
Hasty Defense
popped from stack

Figure IV-6c. Example of Operation Reaction System Operation, Time Step 5

APPENDIX B
IMPLEMENTATION EXAMPLE (CLEW)

Example of Operation Reaction System Implementation: The Corps Level Electronic Warfare (CLEW) Model.

1. Background

The Corps Level Electronics Warfare model was designed to evaluate the effectiveness of various electronic warfare (EW) sensors and their impact on Corps and Division level decisionmaking. The studies in which it has been used have compared the results of simulations with different mixes of sensor systems and, thus, different levels of intelligence and combat information, to determine the value of a given mix of sensor systems in terms of influencing the course of the battle. The discussion here will concern only the small part of the CLEW model which implemented the Operation Reaction System, and the effects of its operation.

In the CLEW model, the basic maneuver units were battalions, and the terrain resolution was 3.5Km. The scenario was a blue armored division defending against an attempted RED breakthrough by four divisions. The simulation was run in two hour game time sessions, with orders being issued to the units as necessary at each interval by man-in-the-loop commanders. The combat method was a Lanchester Square Law mechanism, modified to account for unit disposition, terrain, operational status, and range. Terrain effects impacted on weapons range and effectiveness as well as unit movement. Rivers, three types of roads, degree of forestation, roughness of terrain, and built up areas were represented. Figures B-1 through B-10 show the CLEW ORS, terrain, and combat/ movement effects.

2. The CLEW Operation Reaction System

The CLEW ORS used the Mission and Operation codes shown in figures B-4 to B-8. While most are self explanatory, the Flank Attack, Close Combat, and Road Movement Operations require further discussion:

a. Flank Attack: This operation was designed to represent aspects of a granular attack, in which units would attempt to avoid enemy units and move around their flank.

$$\Delta X = T_c \cdot \frac{A}{D} \cdot \alpha \cdot d^{m-1} \cdot Y \cdot K \cdot S$$

ΔX ≡ ATTRITION OVER TIME INTERVAL

T_c ≡ TERRAIN (AND WEATHER) EFFECTS

A/D ≡ ATTACKER-TO-DEFENDER "MATCH UP"

α ≡ FIRE ALLOCATION OVER TARGET ARRAY

d^{m-1} ≡ DISPOSITION, $m = f$ (WEAPON RANGE, TERRAIN)
$d = f$ (OPERATION CODE)

Y ≡ NUMBER OF WEAPONS

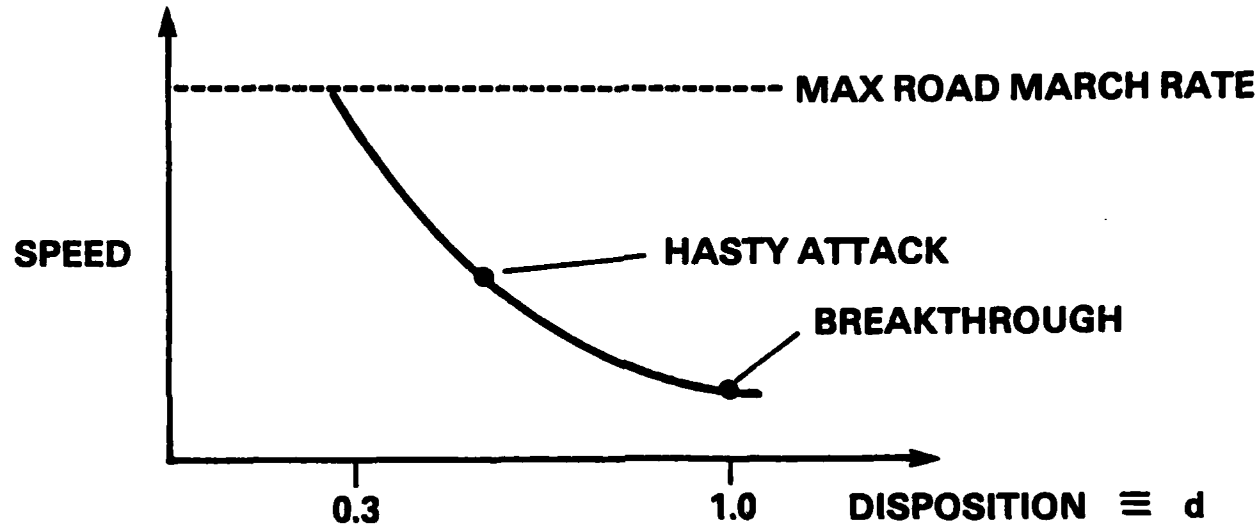
K ≡ KILL RATE (RANGE DEPENDENT)

S ≡ SUPPRESSION EFFECTS (WEAPON & TARGET DEPENDENT)

5613/78W

Figure B-1. CLEW Attrition Algorithm

$$\text{SPEED} = 3 + \frac{1}{d - 0.2} \cdot T_S \cdot F_R$$



$T_S \equiv$ TERRAIN (AND WEATHER) EFFECTS

$F_R \equiv$ FORCE RATION EFFECT

5613/78W

Figure B-2. CLEW Movement Speed

$$\text{"HEX SCORE"} = \underbrace{(5 - F) \cdot \mathrm{COS}\,(D - D^1)}_{\text{DIRECTION CONSIDERATION}} + \underbrace{\frac{Tr}{5} \cdot Sp}_{\text{SPSED}}$$

$$+ \underbrace{(1 - Tr)\,\frac{1}{T_{c/c}}}_{\text{COVER}} + \underbrace{M \cdot U_F \cdot 1}_{\text{MASSING}}$$

$$+ \underbrace{E \cdot U_E}_{\text{ENEMY REACTION}} + \underbrace{C \cdot \frac{1}{(r_{fl}^2 + r_{fr}^2 - 1)}}_{\text{COHESIVENESS}}$$

F ≡ FREEDOM OF DIRECTION

D, D^1 ≡ DIRECTION TO OBJECTIVE, HEX

Tr ≡ TRAFFICABILITY $\quad$ $T_{c/c}$ ≡ TERRAIN COVER & CONCEALMENT

M ≡ MASSING $\quad$ U_F ≡ # OF FRIENDLY UNITS

E ≡ ENEMY "REACTION" FACTOR $\quad$ U_E ≡ # OF ENEMY UNITS

C ≡ COHESIVENESS $\quad$ r_{fl}, r_{fr} ≡ RANGE TO NEAREST "FRIENDLY" (LEFT, RIGHT)

5613/78W

Figure B-3. CLEW Movement Direction Choice Scoring Algorithm

	CASUALTY STATUS	AT OBJECTIVE					NOT AT OBJECTIVE				
		FLANK DANGER		NO FLANK DANGER			FLANK DANGER		NO FLANK DANGER		
		HEX	ADJ	HEX	ADJ	NO CONT.	HEX	ADJ	HEX	ADJ	NO CONT.
NO MTG ENG	–	11	11	3	9	1	11	11	3	2	8
	*	13	14	4	17	1	13	14	4	10	8
	**	12	12	12	5	1	12	12	12	5	8
	***	6	6	6	6	7	6	6	6	6	7
POS-SIBLE MTG ENG	–	15	11	3	9	1	15	11	3	2	8
	*	19	14	4	17	1	19	14	4	10	8
	**	16	12	16	5	1	16	12	16	5	8
	***	6	6	6	6	7	6	6	6	6	7

5613/78W

Figure B-4. Situation Table

ACTION OUTPUT TABLE

#	OPERATION	9	8	7	6	5	4	3	2	1	0
0	PREP. DEF.	0 / 51	0 / 0	0 / 0	61 / 51	51 / 41	41 / 0	0 / 0	0 / 51	0 / 0	0 / 0
1	HASTY DEF.	0 / 51	0 / 0	0 / 0	61 / 51	51 / 41	41 / 0	0 / 41	0 / 51	0 / 41	0 / 0
2	DELAY	0 / 51	0 / 0	0 / 0	61 / 51	51 / 51	51 / 41	41 / 41	0 / 51	0 / 41	0 / 0
3	WITHDRAW	0 / 51	0 / 0	0 / 0	61 / 51	51 / 51	51 / 41	51 / 0	0 / 51	0 / 51	0 / 0
4	HASTY ATK.	0 / 70	0 / 0	0 / 70	61 / 41	41 / 0	70 / 0	0 / 70	0 / 41	0 / 0	0 / 0
5	FLANK ATK.	0 / 70	0 / 0	0 / 70	61 / 41	41 / 0	70 / 0	0 / 70	0 / 41	0 / 0	0 / 0
6	BREAKTHR'H	0 / 70	0 / 0	0 / 70	61 / 41	41 / 0	70 / 0	0 / 70	0 / 41	0 / 0	0 / 0
7	HOLDING ATK.	0 / 70	0 / 0	0 / 70	61 / 41	41 / 0	70 / 0	0 / 70	0 / 41	0 / 0	0 / 0
8	CLOSE CMBT.										
9	RECON	0 / 70	0 / 0	0 / 70	61 / 41	51 / 0	70 / 0	0 / 0	0 / 51	0 / 41	0 / 0
10	ROAD MOVM'T										

5613/78W

Figure B-5. Action Output Table

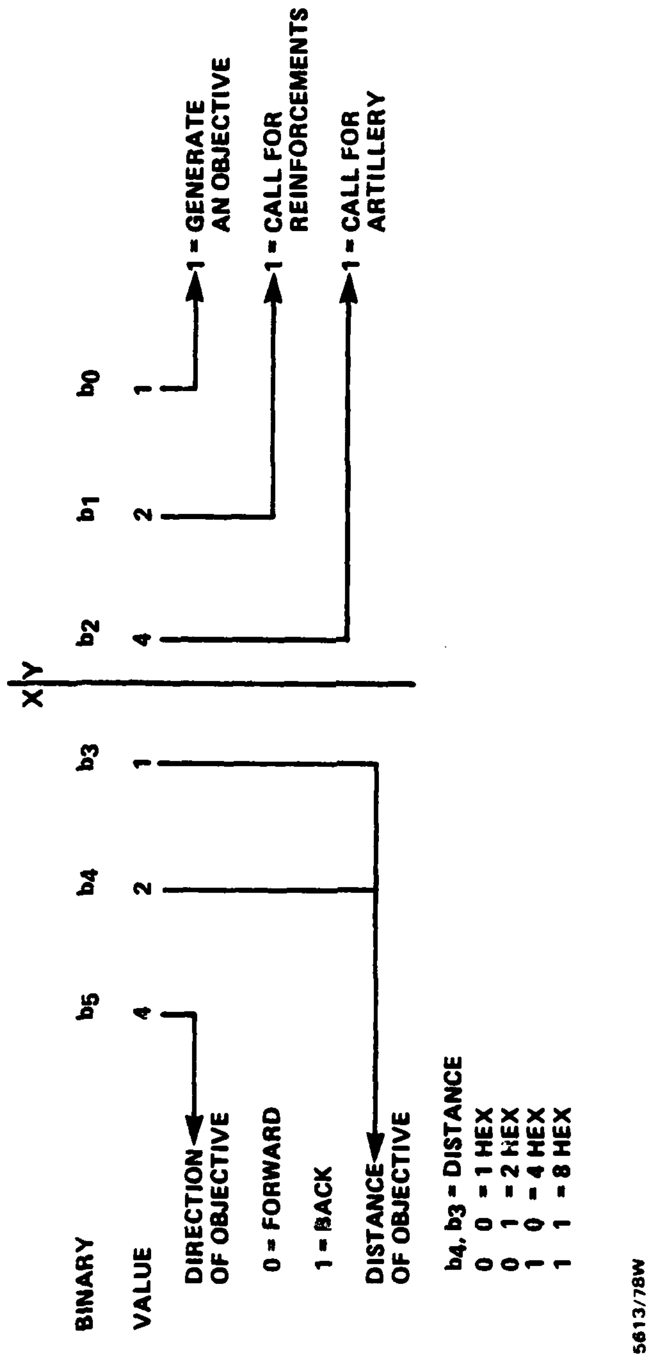


Figure B-6. Action Output Code Translation

BEHAVIOR OUTPUT TABLE

#	OPERATION	9	8	7	6	5	4	3	2	1	0
0	PREP. DEF.	0/4	10/4	0/0	3/4	2/4	2/2	0/2	2/4	0/0	0/0
1	HASTY DEF.	1/4	10/4	1/1	3/4	2/4	2/2	1/2	2/4	1/2	1/1
2	DELAY	1/4	10/4	1/2	3/4	2/4	2/2	2/2	2/4	1/2	2/2
3	WITHDRAW	2/4	10/4	1/2	3/4	3/4	3/3	3/3	3/4	1/3	3/3
4	HASTY ATK.	1/7	10/4	1/7	3/3	7/4	7/4	8/7	4/3	1/4	4/4
5	FLANK ATK.	1/7	10/5	1/7	3/3	7/5	7/5	8/7	5/3	1/5	5/5
6	BREAKTHR'H	1/7	10/6	1/7	3/3	7/6	7/6	8/7	6/3	1/6	6/6
7	HOLDING ATK.	1/7	10/7	1/7	3/3	7/7	7/7	7/7	7/3	1/7	7/7
8	CLOSE CMBT.										
9	RECON	4/3	10/9	1/9	3/3	3/9	2/9	9/9	9/3	4/3	9/9
10	ROAD MOVM'T										

5613/78W

Figure B-7. Behavior Output Table

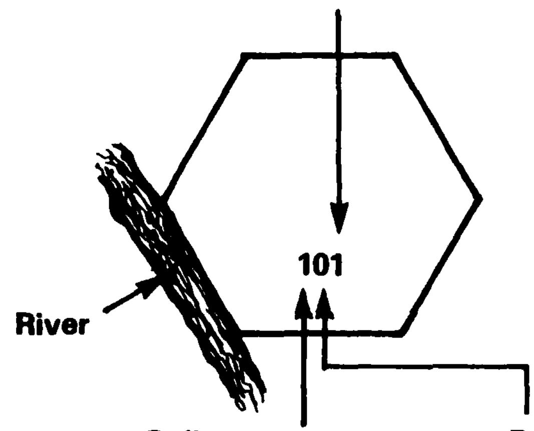
MISSION TRANSITION TABLE

#	OPERATION	9	8	7	6	5	4	3	2	1	0
0	PREP. DEF.	0/1	0/1	0/0	3/1	2/1	1/1	0/1	0/1	0/0	0/0
1	HASTY DEF.	1/1	1/1	1/1	3/1	2/1	1/1	1/1	1/1	1/1	1/1
2	DELAY	2/2	2/2	2/2	3/2	2/2	2/2	2/2	2/2	2/2	2/2
3	WITHDRAW	1/3	3/3	1/2	3/3	3/3	3/3	3/3	3/3	1/3	3/3
4	HASTY ATK.	7/4	4/4	1/4	3/7	7/4	4/4	4/4	4/7	1/4	4/4
5	FLANK ATK.	7/5	5/5	1/5	3/7	7/5	5/5	5/5	5/7	1/5	5/5
6	BREAKTHR'H	7/6	6/6	1/6	3/7	7/6	6/6	6/6	6/7	1/6	6/6
7	HOLDING ATK.	1/7	7/7	1/7	3/7	7/7	7/7	7/7	7/7	1/7	7/7
8	CLOSE CMBT.										
9	RECON	4/9	9/9	1/9	3/9	9/9	9/9	9/9	9/9	4/9	9/9
10	ROAD MOVM'T										

5613/78W

Figure B-8. Mission Transition Table

Terrain Roughness
1 = terrain slope avg>.03 overall or≈15% hills or rugged terrain
2 = terrain slope avg>.06 overall or≈40% hills or very rugged terrain
3 = terrain slope avg>.1 or most of hex impassable to vehicles



Built up extent
1≈15%
2≈40%
3≈70%

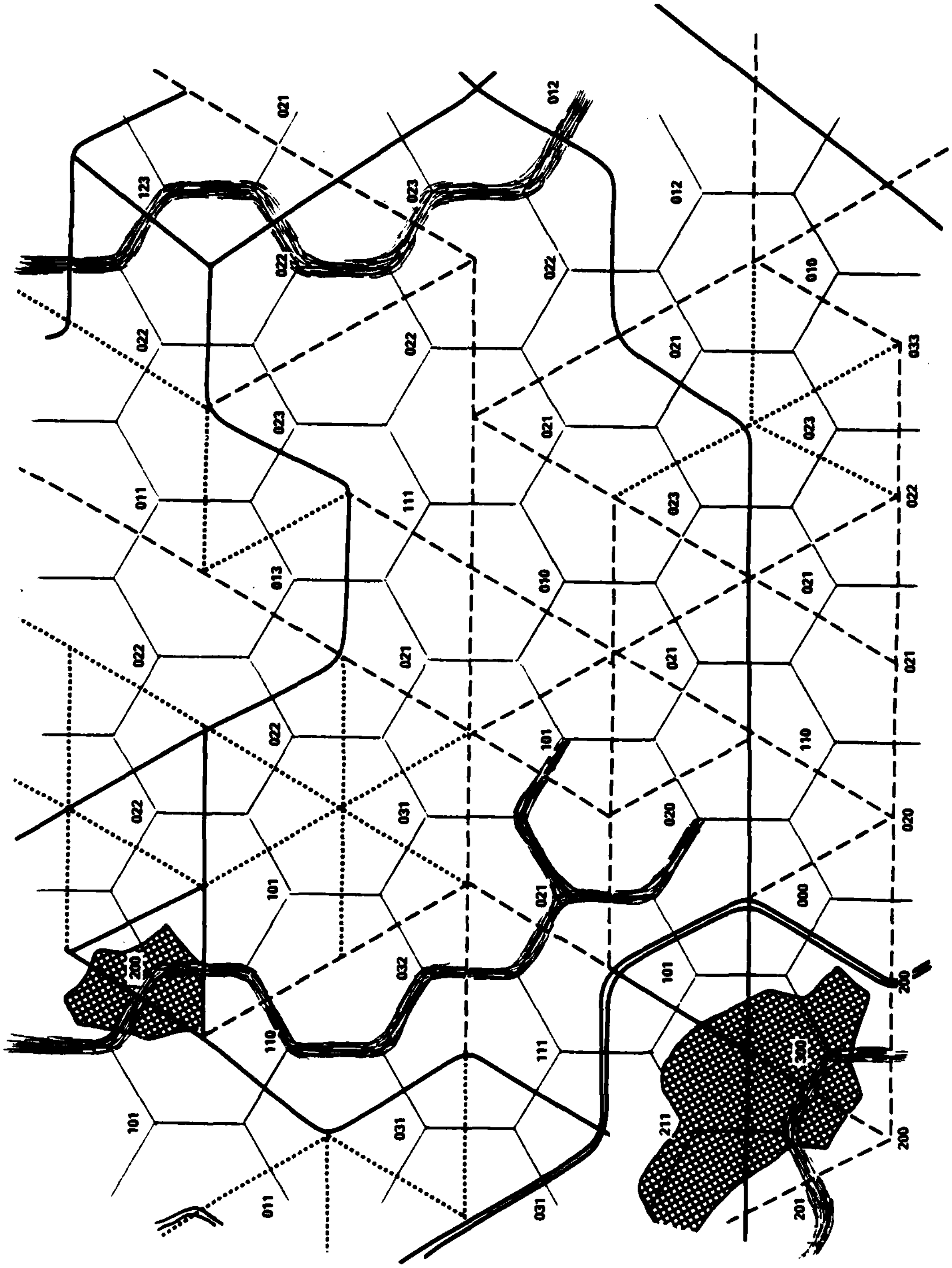
Forrestation
1≈15%
2≈40%
3≈70%

Roads:
Roads do not always correspond one to one with actual highways, but rather indicate the extent to which two hexes are connected.

Autobahn: ══════
Primary: ──────
Secondary: — — — -
Tertiary: ············

5613/78W

Figure B-9. CLEW Map Key



5613/78W

Figure B-10. Sample CLEW Terrain

b. Close Combat: The various attack operations were designed to represent the phase of an attack during which the attacker approaches and attempts to close with the defender. During that time he is moving frequently, and is relatively exposed, while the defender is less exposed, and his firing positions have not all been detected. If the attacker succeeds in closing, however, the situation changes. The defending weapons would be more easily detected, and the weapons effectiveness of both sides would be increased. The Close Combat operation was designed to represent this situation. After an attacking unit is in the same hex as the defender for one ORS cycle, it will normally have an operation output of 'Close Combat'.

c. Road Movement: This operation code is never assigned as a mission. Units not at their objective and not in contact will normally move in this mode.

The CLEW ORS cycle is normally executed after combat and situation evaluation every five minutes. In the case where a unit moves into a new hex, the presence of an enemy unit in the hex is not considered until after a combat cycle has occurred. This allows correct usage of the 'Close Combat' operation.

The inputs to the CLEW ORS are as follows:

a. At Objective: Whether the unit is at the hex given in his operation order as an objective.

b. In Flank Danger: This indicates whether the unit perceives a danger of enemy units moving behind its flanks. A weighted sum of enemy mass, speed, direction and location considerations is compared with a threshhold to determine whether a flank threat is present.

c. Enemy Adjacent: This indicates whether the unit is in contact with enemy units in its own or adjacent hexes.

d. Same Hex: This indicates that an enemy unit is either in the same hex or moving into the game hex, representing an attack during which intense combat is expected.

e. Meeting Engagement: This special condition indicates that the situation is a meeting engagement if the unit is in a defensive

mission. This results when the unit moves either into a hex containing an enemy unit or into which an enemy unit is moving.

f. Effectiveness Level: This condition indicates the casualty status of the unit. A * indicates about 20% casualties, ** for 30% to 40%, and *** indicates an ineffective unit with over 40% to 50% casualties. The exact breakpoints depend on the operation code.

3. <u>Examples</u>

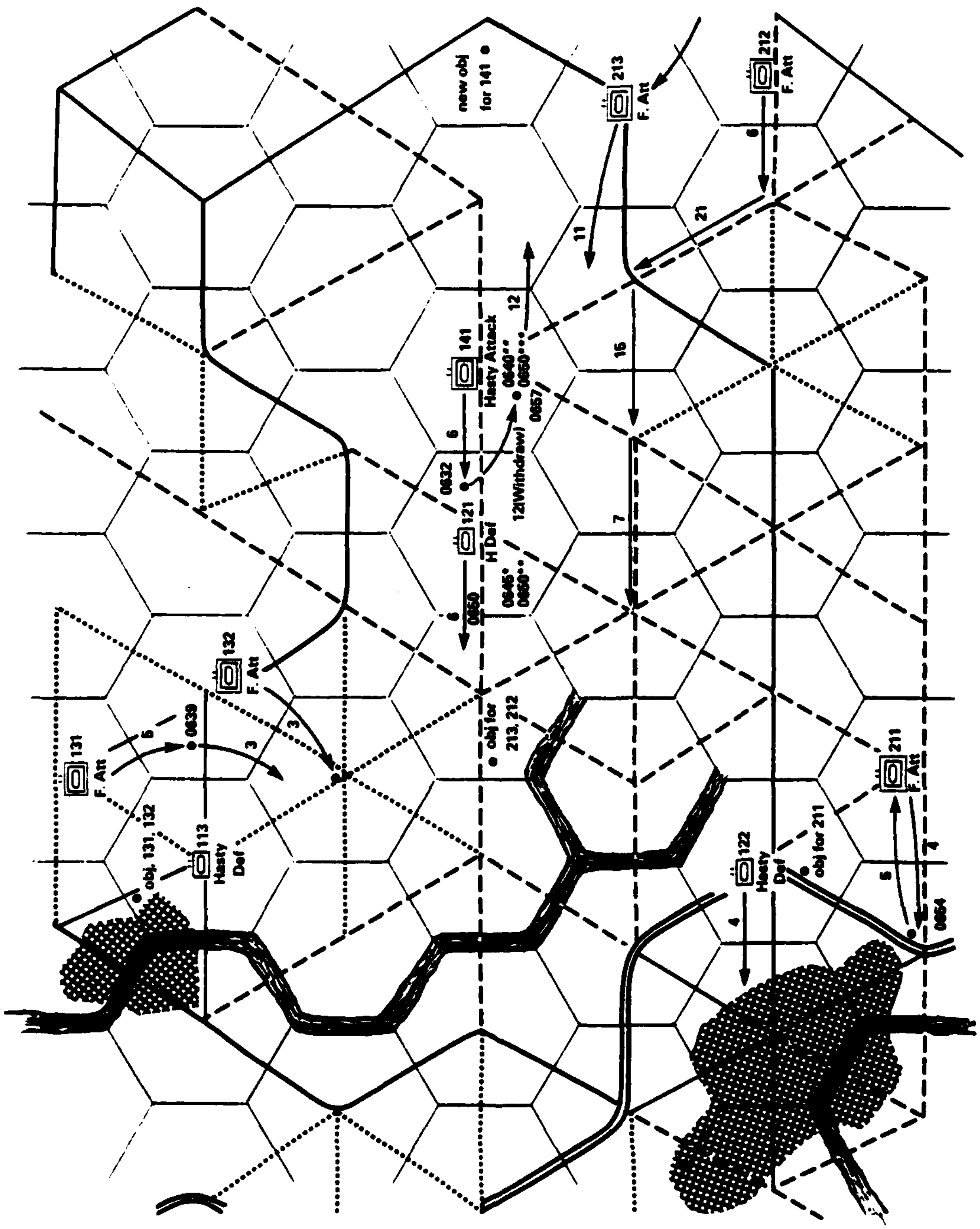
The first examples will be test cases to illustrate the manner in which the ORS works. Figures B-11 and B-12 illustrate a two hour sequence of events which are discussed below:

0600-0700:

(1) Initial positions and orders are indicated on the map. Red units 131 and 132 are given orders to move to the hex containing Blue 113, but their 'Flank Attack' mission code causes them to choose an indirect direction to avoid the Blue unit. The southern direction was chosen by 132 to avoid 131, and by 131 due to the better road (units look ahead only one hex). Unit 113, at its objective, remains in place.

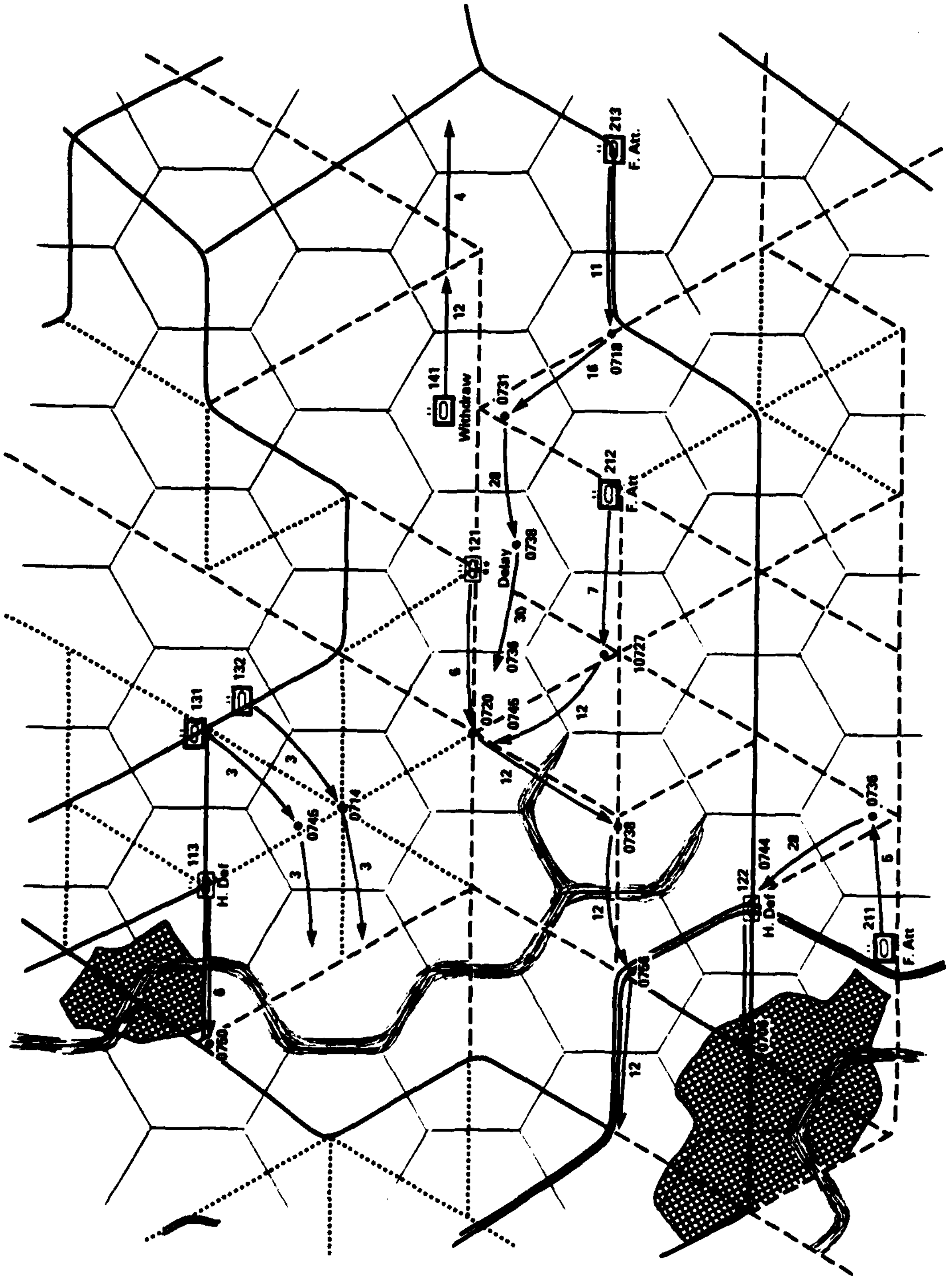
(2) Red unit 141, with a hasty attack order into the hex containing Blue 121, does so. Shortly after arriving in the objective hex, 141 suffers high attrition, and withdraws. 121 also suffers significant casualties and, since the ORS does not consider that the enemy unit in the hex has been defeated, delays back one hex. Red units 212 and 213, which are not in contact, utilize road movement to go in the direction of their objective. At 0644 unit 212 chooses to leave the road rather than risk collision with withdrawing 141.

(3) Blue unit 122 was given a hasty defense order, and Red 211 was ordered to make a flank attack with the objective of Blue 122's starting hex. Rather than directly attack, Red 211 chose the nearest direction not occupied by Blue. Blue 122 detected this movement, and perceived it to threaten its flank, and, as a result, began delaying back one hex. When Red arrived, Blue was



5613/78W

Figure B-11. CLEW Example 1, 0600-0659



5613/78W

Figure B-12. CLEW Example 1,0700-0800

in the process of moving. Rather than move into a hex into which Blue was moving or was occupying, Red 211 chose the nearest available direction toward its objective, which now lay to the NE.

0700-0800:

(1) Red units 131 and 132 continued to flank Blue 113, resulting in the Blue unit's withdrawal across the river.

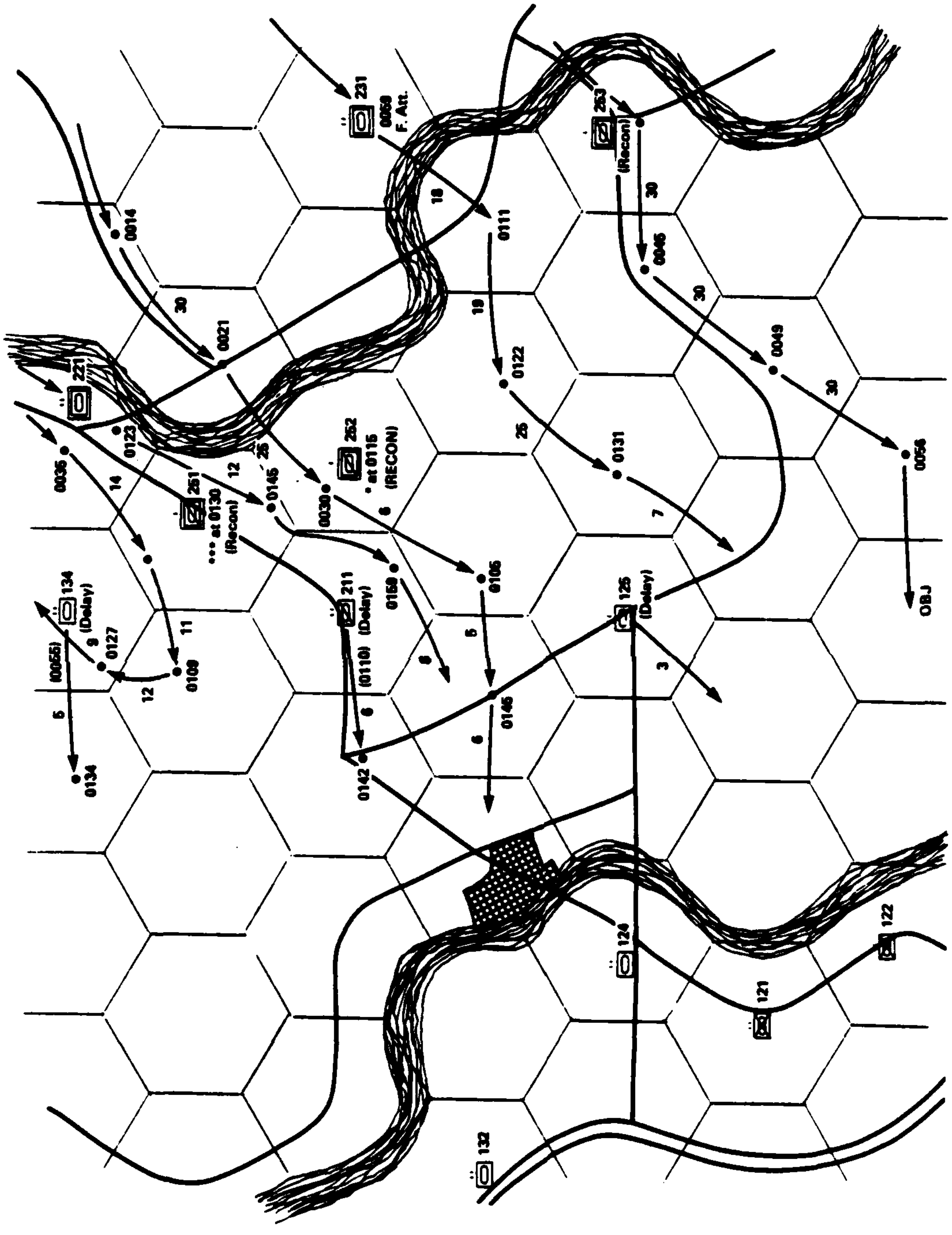
(2) Blue 121, having marginal effectiveness, continued to withdraw. At 0720, when it arrived at its first generated objective, it perceived Red 132 behind its left flank, and so withdrew SW, then West. Red units 213 continued to move using the road movement operation, and did not become engaged. Red 212 arrived at its objective too late to force a meeting engagement with Blue 121.

(3) After Blue 122 arrived at its new objective, Red 211 arrived back at its starting position. At that time it was not in contact, and so was able to use road movement to move directly to its objective, threatening Blue 121.

The second set of examples from CLEW were taken from runs using actual scenario data. A number of adjustments had also been made to the movement and direction algorithms, in particular causing the units to move to an adjacent objective hex. Figures B-13 to B-15 illustrate a sequence of events from a covering force action. Secondary and tertiary roads are not shown.

4. Observations

These examples illustrate the general manner in which units moved during the CLEW data runs, although unit density was often much higher. In most situations, combat proceeded as a series of short high intensity battles as Red units made attacks, separated by periods of maneuver, opportunity five, or relative inactivity. There were, however, numerous occasions in which units did not react as would be expected, showing the need for further development and refinement of this first implementation of the Operation Reaction System.



5613/78W

Figure B-13. CLEW Example 2,0000-0159

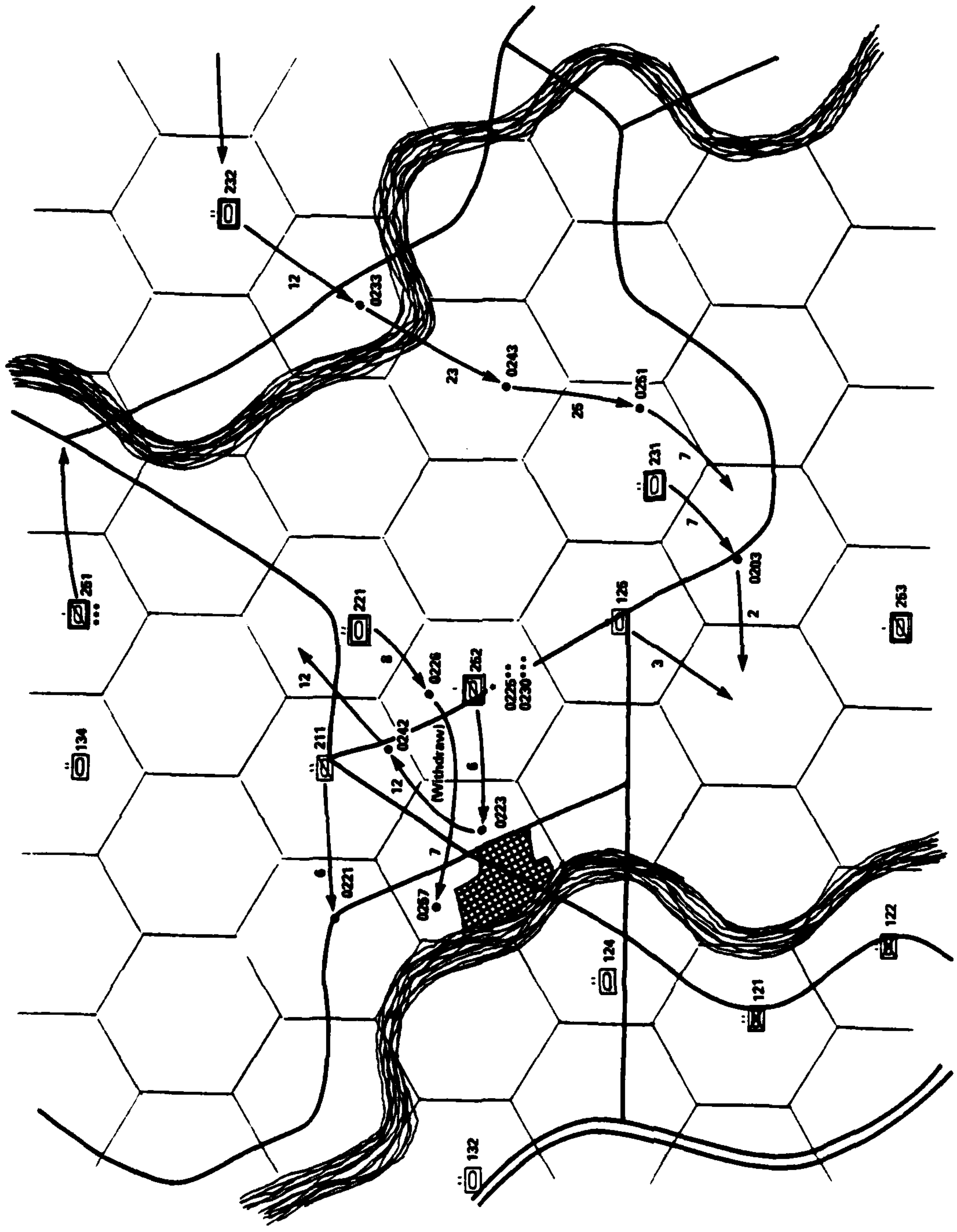
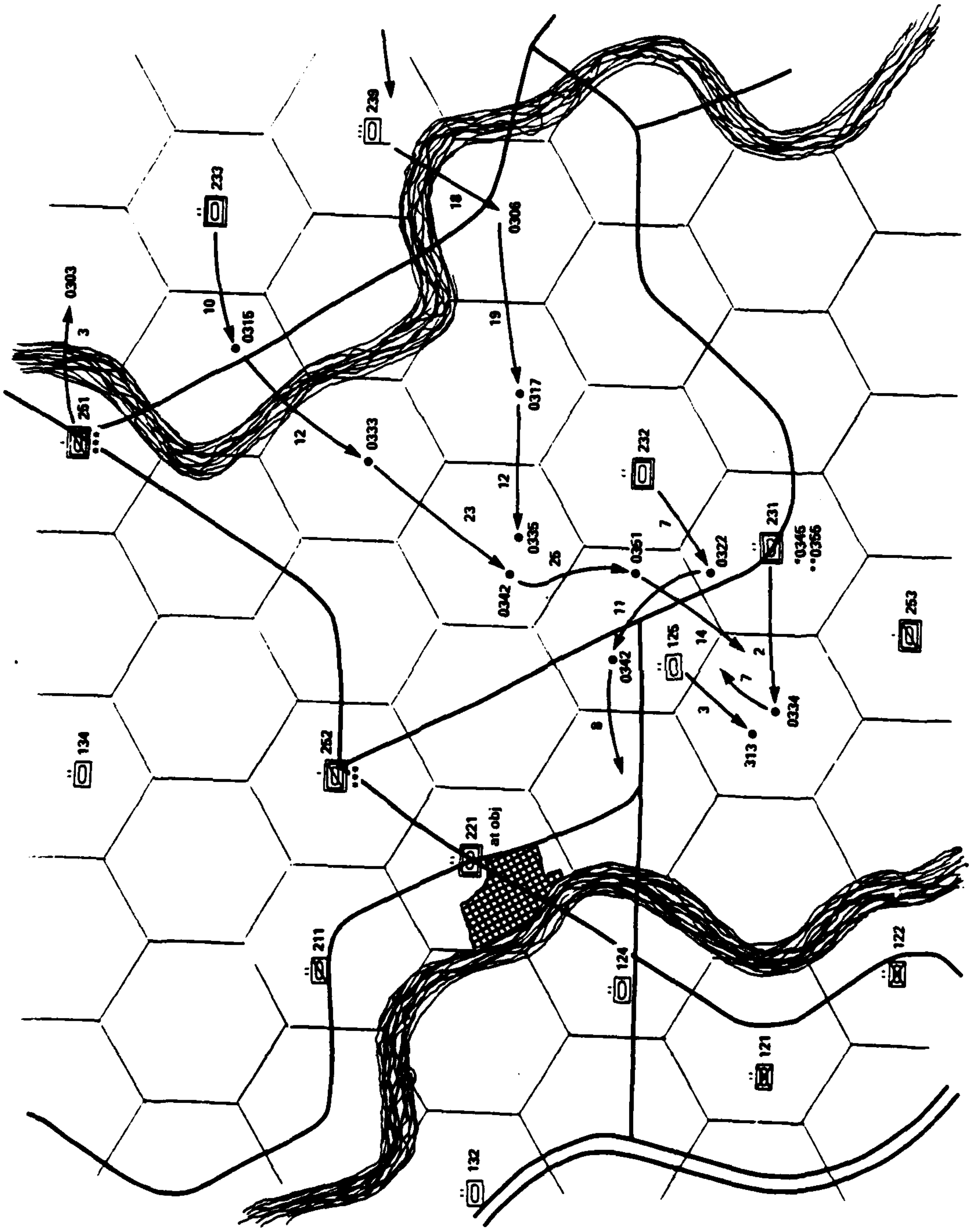


Figure B-14. CLEW Example 2,0200-0259

5613/78W



5613/78W

Figure B-15. CLEW Example 2,0300-0359

APPENDIX C
INPUT LANGUAGE SPECIFICATION

This appendix outlines a method of describing inputs for automatic Operation Reaction System (ORS) table definition and compilation. The generation sequence and the automatic table generation control statements are detailed below:

1. Input Definition

The Operation Reaction System input is the evaluation of a unit's current situation. This is composed of a number of components, each residing in some portion of a situation description word. It is necessary to define those components which are used by the ORS. Situation components can be of two types:

(1) Basic Components: These are bit(s) taken directly from the unit's situation word.

(2) Intermediate Components: These components are functions of the Basic components, and are used to combine basic component in order to reduce the final size of the tables. The applicable statements are as follows:

INPUT (Comp ID) = (Expression)

Where (Expression) = I(Bit#, no.) where Bit# is the bit identification of the location of a basic component in the situation word, and no is the number of values it can assume.

or (Expression) = F(Comp ID, Comp ID_2. . . Comp ID_n) where F is a function of input components which have been previously defined.

EXAMPLES:

INPUT C = I(Ø,2)	Define 'in contact' input
INPUT OBJ = I(1,2)	Define 'at objective' input
INPUT DNGFL = I(2,2)	Define 'danger, left flank' input
INPUT DNGFR = I(2,3)	Define 'danger, right flank' input
INPUT DNGF = DNGFL.OR.DNGFR	Define 'danger on flank' input

2. Operation Definition

This statement defines the operation outputs, each of which are associated with movement, combat, etc. parameters. Since the parameters are not actually part of the ORS and would vary from model to model, they would be inputted separately.

OPERATION (Operation ID) = (Operation code)

EXAMPLES:

OPERATION ATT = 1	Attack operation, assigned code 1
OPERATION DEF = 2	Defend operation, assigned code 2
OPERATION DEL = 3	Delay operation, assigned code 3
OPERATION MCH = 4	March operation, assigned code 4

3. Mission Definition

This statement defines the various mission types to be included. While they may correspond to operation types, they are separate since the mission codes are associated with ORS states while the operation codes are outputs.

MISSION (Mission ID) = (Operation ID)

Operation ID gives a default operation output for this mission.

EXAMPLES:

MISSION ATT = ATT	Attack Mission
MISSION DEF = DEF	Defend Mission
MISSION DEL = DEL	Delay Mission

4. Mission Class Definition

This optional statement allows different missions to be grouped together in a class, so that a given response can be assigned to all members of the class together. Each member in the class can be assigned an index number(s) which can permit multiple transitions from one set of classes to another set of classes to be defined at once. This is particularly valuable in defining multiple doctrines.

CLASS (Class ID) = (Mission ID_1, M_1), (Mission ID_2, M_2). . .

Where Class ID identifies the class of mission

Mission ID; identifies a particular mission code previously defined in a MISSION statement.

M_i gives an index number to the mission. This entry may be omitted.

EXAMPLE:

CLASS	DEF = DEF, DEL	Defines a defensive class which includes the delay and defense missions, without indexing.

5. Action Definition

These statements are used to define action outputs which the ORS can generate. Each possible action is indicated by a field of bits in an action output word.

ACTION (Action ID) = (Bit#, value)

where (Action ID) identifies the particular action

Bit# identifies the location of the action identifier in the output word

and value is a number in octal (B) or decimal giving the value of the action output

or ACTION (Action ID) = (Action ID_2)+(Action ID_3)+. . . . (Action ID_n)

This statement identifies the action ID_1 as being equivalent to the aggregate of the actions listed. If more than one contain the same Bit#, the value for the last is used.

EXAMPLES:

ACTION	PUSH = (0,1)	Push stack
ACTION	POP = (1,1)	Pop stack
ACTION	GOBJ = (2,65B)	Generate objective = 65B (2 hexes in reverse direction in BDM's hexagonal coordinate system
ACTION	PSHG = PUSH + GOBJ	This action both pushes the stack and generates a new objective.

6. Situation Response Definition

This statement is the one which actually specifies the ORS tables. It defines a set of conditions, and the response for those conditions. The statements are processed in order, and immediately applied to the tables. This allows later statements to redefine parts which were also defined in previous statements, adding more detail. Thus, the tables change from statement to statement as this portion of the input is processed. The initial tables contain no actions, and each mission code transitions to itself and outputs a corresponding output operation code (if specified).

IF (Situation Expression) THEN (Output and Transition Expression)

where

(Situation Expression) = (F(Condition$_1$, Condition$_2$, . . . , Condition$_n$))

F is a logical function of the various conditions

(Condition$_i$) = (Comp ID = value)

or (MISSION = Mission ID) or (M=Mission ID)

or (CLASS = Class ID) or (CL = Class ID)

or (CLASSNDX = Class ID) or (CX = M)

This defines the conditions under which the effects are to be implemented, and thus the locations in the ORS tables.

(Output and Transition Expression) = (Result$_1$, Result$_2$, . . , Result$_k$)

(Result$_k$) = (MISSION = Mission ID) or (M=Mission ID)

or (ACTION = Action ID) or (A=Action ID)

or (OPERATION = Operation ID) or (OP=Operation ID)

or (OPERATION = Operation code) or (OP = Operation code)

This specifies the table entries for the three tables.

EXAMPLES:

```
IF (C=Ø.AND.M=ATT) THEN (OP=MCH)
IF (C=Ø.AND.M=DEL) THEN (OP=MCH)
IF (OBJ=1) THEN (OP=DEF,M=DEF)
IF (OBJ=1,M=DEF) THEN (A=POP)
IF (DNGF=1) THEN (OP=DEL,M=DEL)
IF (DNGF=1, M=ATT) THEN (OP=DEF,M=DEF)
IF (DNGF=1, M=ATT,OBJ=Ø) THEN (OP=DEF,M=DEF,A=PSHG)
IF (DNGF=1,M=DEF) THEN (A=GOBJ)
```

This set of situation response definitions defines the table shown in the text as Figure I-3.

APPENDIX D
OPERATIONAL INTELLIGENCE AND PERCEPTION

One function of the maneuver units is the acquisition of intelligence. This information is used directly by them for resolution of combat and the situation evaluation process. The same information is used by the C^2 elements in their reaction and planning.

At each interval in the entity's combat/movement/reaction cycle each entity searches its own and all adjacent hexes. It detects and subsequently may engage targets in all of these hexes, subject to weapons characteristics. The units found can be put into the unit's data base.

After all of the subordinates of a C^2 element, or player, have completed their perceptions, the data base formed by them is evaluated. Enemy units can be put in an intelligence list or information about them passed to the unit's superior.

The information assembled during the perception process must be evaluated for its meaning to the unit. Specific conditions must be recognized so that the unit can respond. This process is basically similar for entities and the players, but differs in specifics.

1. Entity Situation Evaluation

The maneuver units, or entities, evaluate the situation for the following specifics:

(1) Combat Status - whether enemy units are in the same hex, adjacent, or not present.

(2) Force ratio - whether the local ratio of enemy to friendly strength exceeds some operation dependent constant. Similar information could be derived from the unit's own attrition rate instead.

(3) Flanking Threat - whether or not enemy units pose a significant flanking threat

(4) Meeting Engagement or Attack Condition - whether or not the unit is moving into the same hex occupied by an enemy unit or is moving into the same hex into which an enemy unit is also moving, and which is not occupied by friendly units.

(5) Effectiveness - a unit is classified as being in one of three effectiveness states, determined primarily by casualties and with consideration of supply status.

(6) Other - in addition, the presence of a chemical or nuclear attack is noted.

2. <u>Player Situation Evaluation</u>

The C_2 element must evaluate a larger number of conditions, as it must be able to consider maneuver of its component subordinates as well as movement as a whole.

(1) Combat Status, Flanking Threat, and Meeting Engagement/Attack conditions are similar to those for the entities.

(2) Line Integrity - whether the unit's position is intact or not.

(3) Flank Security - whether the command's units are in contact with friendly forces or a secure area (sea or possibly a neutral border or impassable terrain) on either flank.

(4) Angular Moment of Enemy Force - This indicates whether the enemy threat is massed significantly more on either flank.

(5) Effectiveness - this will be an overall effectiveness rating at the command based on those of its component entities.

(6) Nuclear/Chemical Environment - This indicates the nature of the use of threat of use of these weapons.